



Orchestration stigmergique de systèmes multi-agents

LLM : conception et déploiement d'un framework adaptatif

Abdelatif DJEDDOU

Thesis Master in PGE, Master Management des Systèmes d'Informations et des Datas (EMLV)

Date : 6 février 2026

Supervisor : Hajar Kefi

Author's Declaration of Originality

[Include this statement at the front of your thesis document]

I hereby certify that I am the sole author of this thesis and that it presents mainly my ideas, analyses, and evaluation of the findings of my research. This work is the result of my personal writing.

I affirm that I have conducted this work with honesty and full respect for the ethical principles of academic research, by informing the reader of my entire research process. This includes a transparent account of all methods, tools, and resources I used, including, where relevant, the recourse to Artificial Intelligence tools, in line with the School's academic standards.

I certify that, to the best of my knowledge, this document does not infringe upon anyone's copyright and that any ideas, methodologies, quotations, or other material from the work of others have been completely and correctly referenced in the body of the work and in the References and/or Bibliography. All citations follow the referencing system adopted by the School.

Any direct quotations from written or verbal data are shown in quotation marks and appropriately referenced. Any part of this document which has been used in previous academic assessments during my program of study has been identified and referenced in accordance with APA regulations. I confirm that there has been no unauthorized assistance from other students, colleagues, or friends during the production of this document. No part of this work comes from any academic support website.

I declare that this is a true copy of my own work and that I accept that it be submitted through the anti-plagiarism software used by the School for confirmation of this fact.

Date : 6 février 2026

Signature :

Table des matières

Author's Declaration of Originality	1
1 Introduction	5
2 Fondements théoriques de la stigmergie	7
2.1 L'émergence du concept : Grassé et le paradoxe de la coordination	7
2.2 Formalisation algorithmique : l'intelligence en essaim	9
2.3 Extension aux systèmes numériques : la stigmergie comme mécanisme universel	10
2.4 Épistémologie stigmergique et cognition distribuée	10
2.5 Stigmergie cognitive et paradigme Agents & Artifacts	11
2.6 Systèmes adaptatifs complexes : Holland et Kauffman	11
3 Coordination multi-agents et systèmes distribués	12
3.1 Agents LLM et systèmes multi-agents : définitions opératoires	12
3.2 Phéromones numériques : de la biologie au calcul	13
3.3 Espaces de tuples et communication générative	13
3.4 Auto-organisation dans les systèmes multi-agents	14
3.5 Stigmergie empirique dans le développement open source	14
3.6 Développements récents : bio-inspiration dans l'orchestration agentique (2023-2026)	15
3.7 Panorama des frameworks et benchmarks	15
4 Limites des approches multi-agents	16
4.1 L'approche Agentless : la simplicité contre la complexité	16
4.2 Framework MAST : comprendre où les MAS échouent	16
4.3 Analyse coût-efficacité et érosion de l'avantage multi-agents	17
4.4 Positions industrielles et principe de parcimonie	18
5 Migration de code et transformation avec les LLM	18
5.1 Migration à grande échelle : la validation industrielle de Google	18
5.2 Coordination multi-agents pour le codage autonome : les leçons de Cursor . .	19
5.3 Traduction cross-language et modernisation legacy	19
5.4 Évolution et refactoring d'API	20
5.5 Transformation du développeur par l'IA	20
6 Gouvernance, auditabilité et conformité	20
6.1 Guardrails pour écologies humain-IA	20
6.2 Perspective principal-agent et orchestration	21
6.3 Conformité réglementaire : EU AI Act Article 14	22

6.4	Meaningful Human Control : tracking et tracing	22
6.5	Dimensions organisationnelles : résistance et acceptabilité	23
6.6	Responsabilité morale distribuée et gouvernance éthique des systèmes swarm	25
6.7	Sécurité des systèmes multi-agents	25
7	Agentic BPM et workflows	26
7.1	Définition formelle de l'Agentic BPM	26
7.2	Agentic Business Process Management Systems	26
7.3	Du RPA à l'automatisation cognitive	27
7.4	Systèmes adaptatifs complexes appliqués au BPM	27
7.5	Process Mining et LLM pour le cycle de vie BPM	28
8	Benchmarks et évaluation des systèmes multi-agents	29
8.1	MultiAgentBench : évaluation de la collaboration et compétition	29
8.2	REALM-Bench : planification dynamique	29
8.3	Métriques pour la génération de code	29
8.4	Coût-efficacité et reproductibilité	30
9	Cadre conceptuel et identification du gap	30
9.1	Synthèse croisée de la littérature	30
9.2	Gap identifié	30
9.3	Relations conceptuelles entre les blocs thématiques	31
9.4	Complexity Leadership Theory et transformation du rôle managérial	31
9.5	Cadre conceptuel proposé	32
10	Design de recherche	33
10.1	Approche méthodologique	33
10.2	Champ d'étude	33
10.3	Collecte de données	33
10.4	Justification des choix méthodologiques	34
10.5	Analyse des données	35
10.6	Limites anticipées	35
11	Bibliographie	36
A	Annexe IA : usage, comparaison et traçabilité	42
A.1	Pourquoi et comment l'IA a été utilisée	42
A.2	Ce qui a été conservé, modifié, rejeté	42
A.3	Usage MCP amélioré dans le workflow	42
A.4	Outils IA effectivement utilisés	42
A.5	Garde-fous appliqués	43

A.6	Usage Codex pour la production LaTeX	43
A.7	Comparaison de deux outils sur le même prompt	44
A.8	Commentaire critique sur les écarts observés	48

1 Introduction

En 2025, l'intelligence artificielle générative a franchi un seuil décisif dans le domaine du génie logiciel : les grands modèles de langage (LLM) ne se contentent plus de compléter du code ligne par ligne, ils orchestrent désormais des workflows complets de développement, de migration et de maintenance logicielle. Chez Google, 39 migrations de code distinctes totalisant 93 574 éditions individuelles ont été réalisées avec l'assistance de LLM en seulement douze mois, avec 74,45 % des changements appliqués sans modification humaine (Ziftci et al., 2025). Amazon Q Developer transforme automatiquement des applications Java 8 vers Java 17 avec 79 % du code appliqué directement (Amazon, 2024). IBM Watsonx s'attaque à la modernisation de systèmes COBOL, estimant que 775 milliards de lignes de code legacy restent en production à l'échelle mondiale (IBM, 2023).

Ces avancées industrielles convergent vers un constat : la transformation de code à grande échelle ne relève plus de la correction unitaire de bugs, mais de la coordination de multiples agents logiciels opérant simultanément sur des bases de code massives, interdépendantes et critiques pour l'activité des entreprises. Or, l'architecture dominante pour orchestrer ces agents repose sur des modèles hiérarchiques où un superviseur central planifie, distribue et contrôle les tâches de chaque agent subordonné, à l'image des frameworks MetaGPT (Hong et al., 2024), CrewAI ou AutoGen. Cette approche centralisée reproduit, dans le monde logiciel, les structures organisationnelles classiques du command-and-control.

Pourtant, la littérature récente révèle des limitations structurelles préoccupantes de ces architectures hiérarchiques. Le framework MAST identifie 14 modes d'échec distincts des systèmes multi-agents LLM, avec des taux d'échec allant de 41 % à 86,7 % sur sept frameworks état de l'art (Cemri et al., 2025). L'approche Agentless démontre qu'un pipeline séquentiel simple atteint 32 % de résolution sur SWE-bench Lite pour seulement 0,70 USD par problème, surpassant tous les systèmes agents open source (Xia et al., 2024). Gao et al. (2025) observent que les gains de performance des systèmes multi-agents chutent de 10-16 % à seulement 0,8-3 % lorsque des modèles frontières sont utilisés, questionnant la valeur ajoutée réelle de la complexité agentique.

Face à ces constats, une alternative théorique existe : la stigmergie, mécanisme de coordination indirecte où les traces laissées par une action dans un médium stimulent les actions subséquentes (Heylighen, 2016a). Conceptualisée par Grassé (1959) pour expliquer la construction collective des termitières sans superviseur central, la stigmergie a été formalisée par Bonabeau, Dorigo et Theraulaz (1999) comme fondement de l'intelligence en essaim, puis étendue par Heylighen (2016a, 2016b) aux systèmes numériques contemporains tels que Wikipédia, le développement open source et les systèmes de contrôle de version. Ricci et al. (2007) ont proposé le cadre de la *stigmergie cognitive*, démontrant que des agents rationnels dotés de capacités avancées peuvent se coordonner via des artefacts environnementaux manipulables et instrumentables.

L'application de la stigmergie à l'orchestration d'agents LLM pour la transformation de code

présente un intérêt théorique et pratique considérable. D'un point de vue théorique, elle propose de remplacer la communication explicite inter-agents, source de 36,94 % des échecs identifiés par MAST (Cemri et al., 2025), par une coordination médiée par l'environnement. D'un point de vue pratique, les systèmes de contrôle de version comme Git fonctionnent déjà comme des médiums stigmergiques où les commits stimulent les revues, tests et intégrations subséquentes (Bolici et al., 2016). D'un point de vue managérial, la traçabilité intrinsèque des artefacts partagés répond aux exigences croissantes de gouvernance, d'auditabilité et de conformité réglementaire, notamment celles de l'Article 14 de l'EU AI Act imposant une supervision humaine effective des systèmes d'IA à haut risque (Fink, 2025).

Toutefois, la littérature académique présente un gap significatif : si la stigmergie est bien théorisée en biologie et en systèmes distribués classiques, et si les systèmes multi-agents LLM font l'objet d'une recherche active, aucun cadre intégrateur ne propose de combiner coordination stigmergique, agents LLM et gouvernance organisationnelle pour la transformation de code à grande échelle. Les travaux existants traitent ces dimensions de manière cloisonnée : la littérature sur la stigmergie ignore les LLM, la littérature sur les agents LLM privilégie les architectures hiérarchiques, et la littérature en management des systèmes d'information aborde la gouvernance de l'IA sans considérer les mécanismes de coordination émergente.

Ce constat motive la problématique suivante :

Comment une orchestration multi-agents à coordination stigmergique peut-elle répondre aux limites des architectures hiérarchiques pour des workflows complexes de transformation de code, tout en assurant traçabilité, maîtrise des coûts et adoption organisationnelle ?

Cette problématique se décline en trois questions de recherche :

- Mécanisme : Quels types de « phéromones numériques » et règles locales sont adaptés à l'orchestration d'agents LLM pour la migration de code ?
- Performance : Dans quelles conditions la coordination stigmergique surpassé-t-elle une orchestration hiérarchique en termes de taux de succès, de coût et de robustesse ?
- Gouvernance : Comment garantir l'auditabilité et la conformité réglementaire d'un système à coordination émergente ?

La revue suit une progression continue : d'abord les fondements de la stigmergie, puis les mécanismes de coordination, les limites des architectures multi-agents, les usages concrets en migration de code, et enfin les enjeux de gouvernance, de processus et d'évaluation. L'objectif est de construire un cadre cohérent, à la fois techniquement crédible et pertinent pour un public de management des systèmes d'information.

2 Fondements théoriques de la stigmergie

La stigmergie constitue le socle conceptuel de cette recherche. Pour poser ce socle, on part de son origine en biologie, puis on suit son passage vers les systèmes numériques contemporains.

2.1 L'émergence du concept : Grassé et le paradoxe de la coordination

Le terme « stigmergie » fut introduit par le biologiste français Pierre-Paul Grassé dans son étude pionnière des comportements de construction chez les termites *Bellicositermes natalensis* et *Cubitermes* sp. (Grassé, 1959). Grassé observa un paradoxe fondamental : les termites semblaient travailler indépendamment les uns des autres, sans coordination apparente, tout en produisant des structures architecturales collectives d'une remarquable complexité et cohérence. Ce « paradoxe de coordination » remettait en question les deux théories dominantes de l'époque. L'approche organiciste, inspirée de Spencer et Wheeler, postulait l'existence de propriétés émergentes au niveau de la colonie considérée comme un « superorganisme ». L'approche individuelle, défendue par Rabaud, soutenait que toute apparence de coopération n'était que fortuite (Theraulaz & Bonabeau, 1999).

Grassé résolut ce paradoxe en identifiant un troisième mécanisme : la coordination indirecte médiée par l'environnement. Il forgea le néologisme « stigmergie » à partir des racines grecques *stigma* (marque) et *ergon* (travail), définissant ainsi la stimulation des ouvriers par les performances qu'ils ont eux-mêmes accomplies (Grassé, 1959). Le mécanisme stigmergique fonctionne comme une boucle de rétroaction : une configuration architecturale A déclenche une réponse comportementale R chez un terme, qui modifie l'environnement pour créer une nouvelle configuration A', laquelle déclenche à son tour une nouvelle réponse R'. Comme l'expliquent Bonabeau et al. (1999, p. 18), chaque travailleur crée de nouveaux stimuli en réponse aux configurations existantes, ces stimuli agissant sur le même terme ou sur n'importe quel autre membre de la colonie. Les interactions pertinentes sont donc indirectes, transitant par l'environnement.

La Boucle de Rétroaction Stigmergique chez les Termites (The Stigmeric Feedback Loop in Termites)

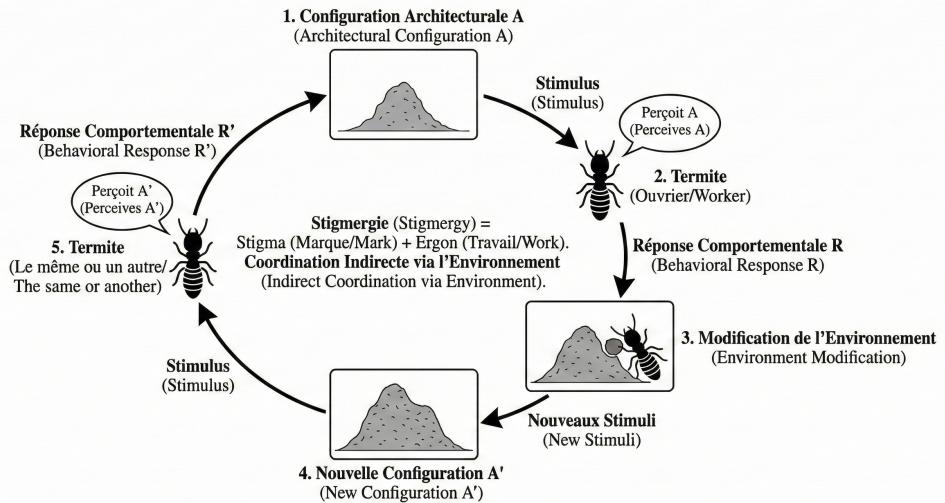


Figure 1 – Boucle de rétroaction stigmergique : une action modifie l'environnement, ce qui déclenche l'action suivante. Source : adaptation à partir de Grassé (1959) et Bonabeau et al. (1999).

Cette découverte établit plusieurs principes fondamentaux pour la compréhension des systèmes auto-organisés. Premièrement, la décentralisation : aucun agent individuel ne possède une vision globale ni n'exerce de contrôle centralisé. Deuxièmement, la construction incrémentale : les agents exploitent ce que d'autres ont construit pour apporter leur propre contribution. Troisièmement, la flexibilité adaptative : lorsque l'environnement change en raison d'une perturbation externe, les agents répondent de manière appropriée sans reprogrammation spécifique (Bonabeau et al., 1999). Quatrièmement, la réduction de la communication : la stigmergie minimise le besoin de communication directe entre agents, l'environnement servant de mémoire partagée distribuée (Marsh & Onof, 2008).

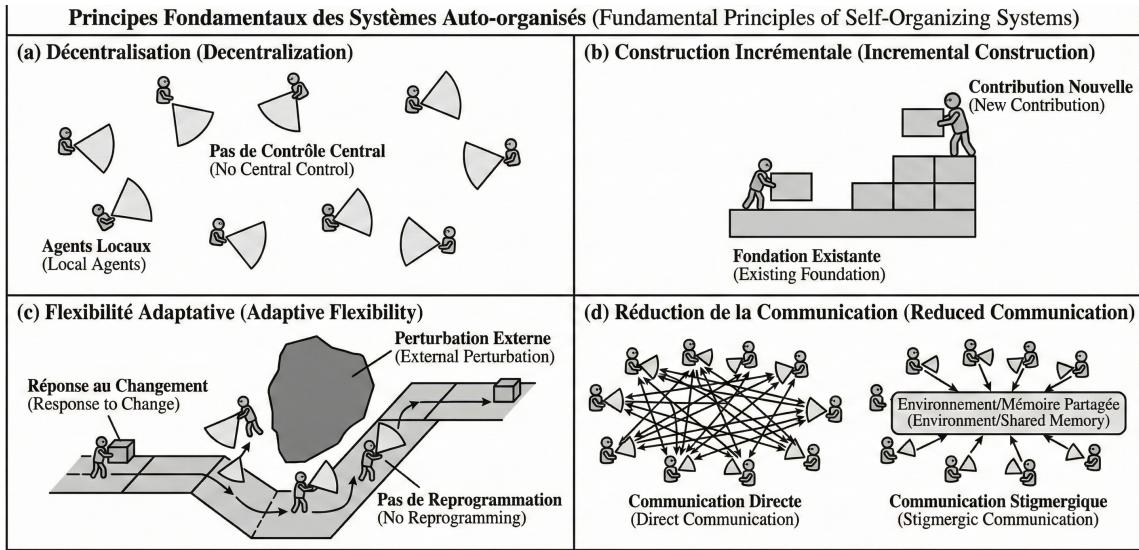


Figure 2 – Principes fondamentaux des systèmes auto-organisés : décentralisation, contribution incrémentale, adaptation et mémoire environnementale partagée. Source : Bonabeau et al. (1999) ; Marsh et Onof (2008).

2.2 Formalisation algorithmique : l'intelligence en essaim

L'ouvrage fondateur *Swarm Intelligence : From Natural to Artificial Systems* (Bonabeau et al., 1999), publié dans la série du Santa Fe Institute, établit l'intelligence en essaim comme discipline scientifique légitime. Les auteurs y démontrent systématiquement comment les principes d'autonomie, de décentralisation et d'émergence peuvent remplacer le contrôle centralisé traditionnel dans la résolution de problèmes complexes. Leur contribution majeure réside dans l'articulation du rôle de la modélisation comme interface entre la compréhension de la nature et la conception de systèmes artificiels (Bonabeau et al., 1999, p. 18). Cette approche méthodologique permet d'explorer les modèles biologiquement motivés au-delà de leurs contraintes naturelles initiales, par exemple, les algorithmes d'optimisation par colonie de fourmis (ACO) utilisent des taux de décroissance de phéromones biologiquement implausibles mais computationnellement efficaces.

Parallèlement, Theraulaz et Bonabeau (1999) clarifient dans leur synthèse historique parue dans *Artificial Life* la conceptualisation de la stigmergie en distinguant deux mécanismes fondamentaux. La stigmergie quantitative repose sur l'intensité de la trace qui affecte la probabilité d'une action : plus une piste de phéromone est concentrée, plus la probabilité qu'une fourmi la suive est élevée (Theraulaz & Bonabeau, 1999, p. 104-105). La stigmergie qualitative implique des stimuli qualitativement différents déclenchant des réponses comportementales qualitativement différentes, comme dans la construction de nids de guêpes où la configuration spatiale détermine le type de réponse architecturale (Theraulaz & Bonabeau, 1999, p. 105). Cette typologie fournit un cadre conceptuel pour guider le choix entre approches de coordination dans la conception de systèmes multi-agents.

2.3 Extension aux systèmes numériques : la stigmergie comme mécanisme universel

Francis Heylighen entreprend, dans sa synthèse en deux parties publiée dans *Cognitive Systems Research*, la première formalisation véritablement générale de la stigmergie (Heylighen, 2016a, 2016b). Il propose une définition unifiée : la stigmergie est le mécanisme de coordination indirecte où la trace laissée par une action dans un médium stimule les actions subséquentes (Heylighen, 2016a). Cette définition capture l'essence bidirectionnelle du processus : l'action (*ergon*) laisse une marque (*stigma*), qui stimule de nouvelles actions, formant une boucle de rétroaction.

Heylighen (2016b) enrichit la typologie de Theraulaz et Bonabeau en distinguant la stigmergie sématectonique, communication via modification physique de l'environnement où la stimulation provient du travail lui-même, et la stigmergie basée sur marqueurs, communication via mécanismes de signalisation qui ne contribuent pas directement à la tâche mais rendent les traces d'activité plus saillantes. En contexte numérique, cette distinction s'illustre avec Wikipédia : les lecteurs sont stimulés à améliorer les pages soit directement en remarquant les lacunes du texte (stigmergie sématectonique), soit indirectement via les notes « à faire » résument les tâches restantes (stigmergie basée sur marqueurs) (Heylighen, 2016b, p. 51).

L'identification par Heylighen (2016a) de la stigmergie comme mécanisme sous-jacent de multiples phénomènes de coordination numérique est particulièrement pertinente pour cette recherche. Wikipédia, le développement open source, les systèmes de contrôle de version et même le mécanisme de prix du marché reposent sur des dynamiques stigmergiques où les actions passées, réifiées dans un médium, stimulent les actions futures. Heylighen (2016a, p. 6) explique pourquoi ce mécanisme est resté longtemps sous-étudié : l'interaction stigmergique étant par définition indirecte, notre esprit biaisé pour rechercher des causes directes tend à négliger la possibilité qu'un agent puisse influencer le comportement d'un autre uniquement via la route indirecte d'une trace laissée dans un environnement passif.

2.4 Épistémologie stigmergique et cognition distribuée

Marsh et Onof (2008) opèrent un pont entre stigmergie et philosophie de l'esprit dans *Cognitive Systems Research*. Partant d'une conception située de la cognition, ils proposent que la stigmergie constitue le mécanisme de coordination fondamental pour comprendre comment la connaissance dans les communautés complexes est essentiellement stigmergique (Marsh & Onof, 2008, p. 136). Leur caractérisation formelle des systèmes stigmergiques comporte trois ensembles de propriétés : un environnement global comprenant des environnements locaux avec une dynamique interne propre et une perceptibilité seulement partielle ; des agents multiples présentant rationalité limitée, comportement auto-organisé, stochastique et adaptatif ; et des propriétés émergentes résultant des interactions, propriétés qui ne sont ni prédictibles ni réductibles à des constituants plus simples (Marsh & Onof, 2008, p. 138).

Cette formalisation est directement applicable aux systèmes multi-agents LLM : les agents disposent d'une rationalité limitée par leur fenêtre de contexte, ils se comportent de manière stochastique (génération probabiliste), et leurs interactions via un dépôt de code partagé produisent des résultats collectifs non réductibles aux contributions individuelles.

2.5 Stigmergie cognitive et paradigme Agents & Artifacts

Ricci, Omicini, Viroli, Gardelli et Oliva (2007) proposent une extension conceptuelle majeure avec la stigmergie cognitive, applicable aux agents rationnels dotés de capacités cognitives avancées, par opposition aux agents réactifs simples des approches biologiques classiques. Leur cadre, publié dans *Environments for Multi-Agent Systems III* (Springer LNCS 4389), établit un pont fondamental entre stigmergie classique et architectures de systèmes multi-agents (MAS) modernes.

Le paradigme Agents & Artifacts repose sur une distinction architecturale clé : les agents (entités computationnelles autonomes et proactives) interagissent indirectement via des artefacts (ressources environnementales manipulables supportant les activités des agents). Les artefacts ne sont pas de simples objets passifs mais des médiateurs actifs de coordination, fournissant des interfaces d'interaction bien définies et rendant la coordination indirecte explicitement instrumentable (Ricci et al., 2007). Contrairement aux phéromones biologiques dont la dynamique est fixée par la nature, les artefacts numériques peuvent être conçus, configurés et adaptés pour répondre aux besoins spécifiques de coordination. Leur état interne peut être exposé, permettant aux agents rationnels d'observer les traces d'actions passées et d'ajuster leur comportement. La durée de vie des traces stigmergiques peut être contrôlée programmatiquement, offrant une alternative à l'évaporation naturelle des phéromones.

Cette contribution s'avère fondamentale pour l'orchestration multi-agents moderne basée sur les LLM, car elle fournit le cadre conceptuel permettant de traiter les artefacts numériques, documents partagés, bases de connaissances, systèmes de tickets, logs d'exécution, comme médiateurs de coordination stigmergique. Les agents LLM, dotés de capacités de raisonnement complexes, peuvent interpréter et manipuler ces artefacts avec une sophistication bien supérieure aux agents biologiques ou réactifs traditionnels. En outre, les artefacts conservent l'historique des modifications, permettant l'audit *a posteriori* des décisions collectives, les artefacts sont inspectables par des opérateurs humains maintenant le *human-in-the-loop*, et la coordination est externalisée dans des artefacts dédiés séparant la logique métier de la mécanique de coordination (Ricci et al., 2007).

2.6 Systèmes adaptatifs complexes : Holland et Kauffman

Les travaux fondateurs de Holland (1995) et Kauffman (1993) fournissent le cadre théorique des systèmes adaptatifs complexes (CAS) permettant de comprendre comment l'organisation complexe émerge d'agents adaptatifs simples. Holland (1995) introduit le concept d'ordre ca-

ché pour décrire comment des patterns organisationnels complexes peuvent émerger de règles d'interaction simples entre agents, démontrant que la complexité structurelle n'est pas nécessairement le résultat d'une conception centralisée explicite. Kauffman (1993) développe le concept d'ordre gratuit et la dynamique au bord du chaos, montrant que les systèmes biologiques et organisationnels opèrent dans une zone critique entre ordre rigide et désordre chaotique où l'adaptabilité est maximale.

Ces fondations théoriques sont essentielles car elles établissent que l'auto-organisation est une propriété fondamentale des systèmes composés d'agents adaptatifs interagissant localement, que la complexité peut émerger de la simplicité, et que les systèmes opérant au bord du chaos peuvent s'adapter dynamiquement aux perturbations sans reconfiguration centralisée. Ces principes fournissent le fondement théorique pour concevoir des systèmes d'orchestration multi-agents capables d'auto-organisation, d'adaptation et de coordination décentralisée.

Ces bases théoriques permettent maintenant de passer à la question pratique : comment transformer cette logique de coordination indirecte en mécanismes réellement utilisables dans des systèmes multi-agents logiciels.

3 Coordination multi-agents et systèmes distribués

La transposition des principes stigmergiques vers des systèmes informatiques distribués demande des mécanismes concrets. On clarifie d'abord les notions d'agent et de système multi-agent, puis on regarde comment la coordination passe par un environnement partagé.

3.1 Agents LLM et systèmes multi-agents : définitions opératoires

Avant de détailler la coordination, il faut distinguer deux niveaux souvent mélangés dans les publications récentes : l'agent LLM et le système multi-agent (Cemri et al., 2025). Un agent LLM est un logiciel qui reçoit des consignes, garde un état (historique ou mémoire) et agit dans un environnement via des outils. Un système single-agent concentre tout dans une seule boucle d'action. Un système multi-agent répartit le travail entre plusieurs agents spécialisés qui se coordonnent entre eux (Cemri et al., 2025 ; Gao et al., 2025).

La littérature avance des bénéfices clairs pour les MAS : découpage des tâches, parallélisation et spécialisation des rôles. Mais ce choix a aussi des coûts : plus de latence, plus de tokens, plus de difficulté de débogage et de gouvernance (Gao et al., 2025 ; Kapoor et al., 2024).

Un constat important pour cette recherche est que les MAS LLM dominants s'appuient sur des patterns de coordination conversationnels (échanges de messages) ou hiérarchiques (superviseur-workers). Rodriguez (2026) formalise une alternative via une coordination par champs de pression et décroissance temporelle, où les agents opèrent localement sur un artefact commun. Ce déplacement vers l'environnement comme support de coordination est cohérent avec la perspective *environment-first* de Weyns et al. (2007), qui conceptualisent l'environnement

comme une abstraction de première classe dans les systèmes multi-agents.

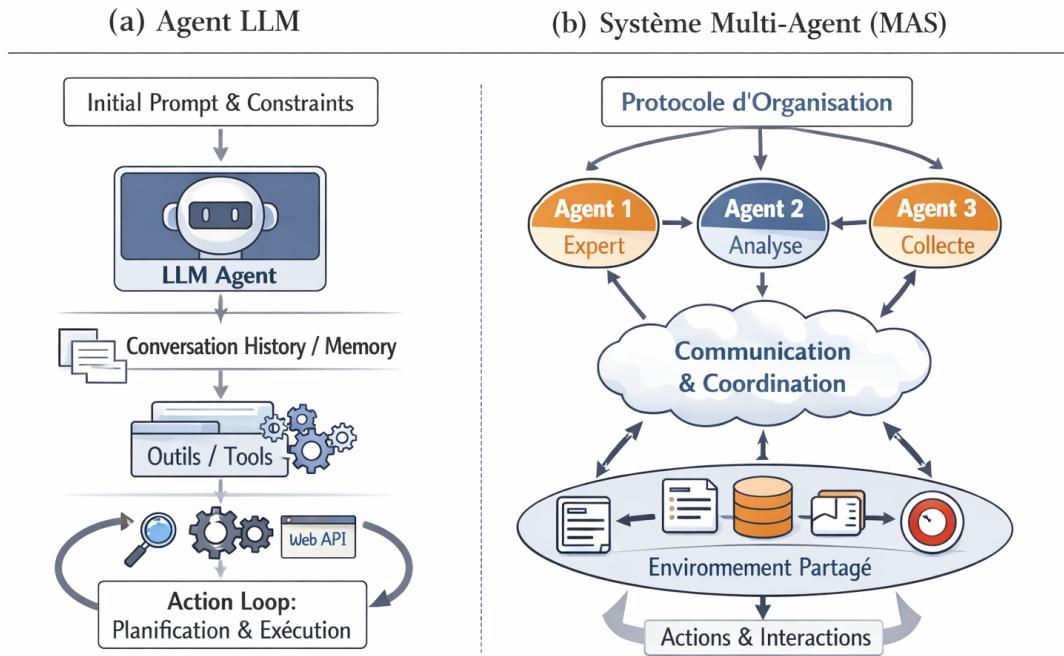


Figure 3 – Agent LLM vs système multi-agent (MAS) : comparaison des niveaux d'organisation et des modes de coordination. Source : adaptation de Cemri et al. (2025) et Gao et al. (2025).

3.2 Phéromones numériques : de la biologie au calcul

Parunak (1997) développe dans son article fondateur une traduction systématique des principes biologiques en principes d'ingénierie pour systèmes multi-agents, identifiant trois principes fondamentaux : localité de perception et d'action, coordination stigmergique, et feedback positif. Il souligne que les agents dans les systèmes naturels sont remarquablement simples, mais que leur interaction produit des comportements collectifs sophistiqués (Parunak, 1997, p. 70). Cette simplicité individuelle couplée à la complexité émergente constitue un avantage architectural majeur : la robustesse du système dépend de la redondance et de l'adaptabilité collective plutôt que de la sophistication de composants individuels fragiles.

3.3 Espaces de tuples et communication générative

Le modèle Linda, introduit par Gelernter (1985), révolutionne la coordination dans les systèmes distribués en proposant la communication générative via un espace de tuples, mémoire associative partagée où les processus déposent et récupèrent des tuples sans connaître l'identité des autres processus. Gelernter (1985, p. 81) critique le modèle classique de message-passing qui crée un couplage temporel et spatial rendant les programmes fragiles. Linda permet une co-

ordination découplée selon trois dimensions : temporelle (producteur et consommateur n'ont pas besoin d'exister simultanément), spatiale (les processus ne connaissent pas l'identité les uns des autres), et de synchronisation (la communication est asynchrone).

L'espace de tuples constitue une concrétisation computationnelle du médium stigmergique théorisé par Heylighen (2016a). Les tuples représentent les traces laissées par les actions des agents, qui stimulent les actions subséquentes d'autres agents. Gelernter (1985, p. 89) observe que les données acquièrent une existence indépendante dans l'espace de tuples, incarnant le principe stigmergique de médiation environnementale.

Carriero et Gelernter (1992) formalisent un principe architectural fondamental : la séparation entre modèle de calcul et modèle de coordination. La coordination doit être traitée comme une préoccupation de première classe, distincte et orthogonale au calcul (Carriero & Gelernter, 1992, p. 98). Cette orthogonalité justifie théoriquement la séparation entre services métier et mécanismes d'orchestration, fondement des architectures de microservices contemporaines.

Mamei et Zambonelli (2009) proposent l'extension TOTA (*Tuples On The Air*) introduisant des tuples distribués capables de se propager de manière autonome à travers le réseau, de s'auto-maintenir face aux changements topologiques, et de se transformer selon leur contexte de propagation. Cette approche génère des champs computationnels, gradients distribués d'information guidant les agents, et s'avère particulièrement pertinente pour les architectures de microservices dynamiques (Mamei & Zambonelli, 2009).

3.4 Auto-organisation dans les systèmes multi-agents

Serugendo, Gleizes et Karageorgos (2005) proposent une taxonomie complète des mécanismes d'auto-organisation, définissant l'auto-organisation comme un processus dynamique et adaptatif où les systèmes acquièrent et maintiennent une structure sans contrôle externe explicite (Serugendo et al., 2005, p. 166). Weyns, Omicini et Odell (2007) complètent cette perspective en proposant l'environnement comme abstraction de première classe dans les systèmes multi-agents, argumentant que les environnements ne sont pas de simples conteneurs passifs mais des entités actives structurant la coordination. Ce positionnement résonne directement avec le cadre stigmergique où l'environnement est le médium de coordination plutôt qu'un simple support d'exécution.

3.5 Stigmergie empirique dans le développement open source

Bolici, Howison et Crowston (2016) fournissent une validation empirique de la coordination stigmergique dans le développement logiciel libre (FLOSS). Leur étude, publiée dans *Cognitive Systems Research*, démontre que les équipes FLOSS utilisent des mécanismes stigmergiques via les systèmes de contrôle de version, les systèmes de suivi de bugs, et les listes de diffusion, où les contributions de code servent de stimuli déclenchant les activités subséquentes de revue, test

et intégration (Bolici et al., 2016). La traçabilité intrinsèque des systèmes de contrôle de version crée automatiquement une trace complète et auditable des modifications.

Cette observation est directement pertinente pour notre problématique : Git, outil omniprésent du génie logiciel, fonctionne déjà comme un médium stigmergique. Les commits représentent les traces, les branches les chemins de phéromones, les pull requests les points de validation collective, et les tests CI/CD les mécanismes de feedback environnemental. L'orchestration multi-agents stigmergique ne nécessite donc pas la création d'une infrastructure nouvelle, mais l'instrumentation et l'extension d'outils déjà adoptés industriellement.

3.6 Développements récents : bio-inspiration dans l'orchestration agentique (2023-2026)

Un développement notable du SOTA 2023-2026 est la réintroduction explicite de mécanismes d'intelligence en essaim dans l'orchestration agentique. Chari et al. (2025) proposent ACO-ToT, combinant LLM et optimisation par colonie de fourmis pour apprendre des chemins de raisonnement via renforcement de phéromones dans un arbre de pensée. Zhang et al. (2025) introduisent SwarmAgentic, un cadre visant à automatiser la conception de systèmes multi-agents via une population de systèmes candidats évoluant par mises à jour guidées par feedback, inspirées du Particle Swarm Optimization. Rodriguez (2026) formalise une coordination émergente par champs de pression et décroissance temporelle, où les agents opèrent localement sur un artefact commun.

Ces travaux convergent vers le positionnement de cette recherche : le déplacement d'une partie de la coordination depuis des protocoles conversationnels explicites vers un médium partagé et des signaux d'évaluation, favorisant l'adaptation et réduisant l'overhead d'orchestration. Rahman et al. (2025) interrogent toutefois la transposition du terme « *swarm* » aux frameworks LLM, soulignant les différences fondamentales entre agents biologiques simples et agents LLM complexes dotés de capacités de raisonnement avancées.

3.7 Panorama des frameworks et benchmarks

Deux familles complémentaires de frameworks coexistent pour concevoir des systèmes multi-agents : les plateformes MAS classiques pré-LLM (JADE, Jason, SPADE) reposant sur le message-passing et des modèles organisationnels explicites, et les frameworks LLM-native (LangGraph, AutoGen, CrewAI, MetaGPT) orientés orchestration d'agents LLM intégrant tool-use, gestion d'état et topologies d'interaction (Geng & Chang, 2025 ; Cemri et al., 2025). MetaGPT formalise des *Standardized Operating Procedures* (SOPs) pour réduire les incohérences des chaînes d'agents dans les tâches de génie logiciel (Hong et al., 2024). OpenHands propose une plate-forme open source pour développer et évaluer des agents généralistes capables d'interagir avec dépôts, shell et tests (Wang et al., 2025).

L'essor des agents s'accompagne de benchmarks dédiés : AgentBench vise l'évaluation « LLM-as-Agent » dans plusieurs environnements (Liu et al., 2024), SWE-bench structure l'évaluation sur des issues GitHub réelles (Jimenez et al., 2024), et SWE-agent met en avant l'importance des interfaces agent-ordinateur (Yang et al., 2024). Ces benchmarks sont repris plus loin dans la section dédiée à l'évaluation.

Ce cadrage permet d'aborder le point suivant de manière plus lucide : même si les MAS promettent beaucoup, ils ont aussi des limites concrètes qu'il faut intégrer dès la conception.

4 Limites des approches multi-agents

Pour garder une lecture équilibrée, il faut aussi regarder les limites des architectures multi-agents et les cas où des approches plus simples fonctionnent mieux.

4.1 L'approche Agentless : la simplicité contre la complexité

L'article Agentless (Xia et al., 2024) constitue une contribution majeure remettant en question les présupposés de l'ingénierie logicielle basée sur des agents autonomes. Xia et al. (2024) proposent une approche simplifiée structurée en trois phases séquentielles : localisation hiérarchique des emplacements d'édition pertinents, génération de multiples patches candidats en format diff, et validation via tests de reproduction et régression. Contrairement aux outils basés sur agents, Agentless contient des étapes bien définies sans laisser l'agent LLM décider des actions futures ni utiliser d'outils complexes (Xia et al., 2024, p. 4).

Les résultats sur SWE-bench Lite sont remarquables : 32 % de précision (96 corrections sur 300 problèmes) pour 0,70 USD de coût moyen par problème, surpassant tous les systèmes agents open source (Xia et al., 2024, p. 3). Cette performance est d'autant plus significative qu'elle est obtenue sans mécanismes complexes d'agents autonomes. Les auteurs espèrent qu'Agentless aidera à réinitialiser le baseline et le point de départ pour les agents logiciels autonomes (Xia et al., 2024, p. 3).

Il convient de contextualiser cette critique : Agentless est focalisé sur le software engineering avec SWE-bench comme benchmark spécifique. La généralisation à des workflows de migration multi-fichiers à grande échelle, où les dépendances inter-composants créent une complexité structurelle supérieure, nécessite une validation empirique supplémentaire.

4.2 Framework MAST : comprendre où les MAS échouent

Le framework MAST (Cemri et al., 2025) est utile parce qu'il montre clairement les familles d'échecs récurrents dans les systèmes multi-agents LLM. En pratique, ces échecs reviennent souvent à trois problèmes : des tâches mal cadrées, des agents qui se comprennent mal entre eux, et des sorties insuffisamment vérifiées avant la fin du processus.

Même sans entrer dans les détails méthodologiques, le message est fort : les taux d'échec observés restent élevés sur plusieurs frameworks open source, ce qui indique un enjeu de conception, pas seulement de puissance de modèle. Pour un lecteur orienté management SI, cela renforce l'idée qu'un bon système ne dépend pas uniquement du modèle, mais surtout de l'architecture de coordination et des mécanismes de contrôle.

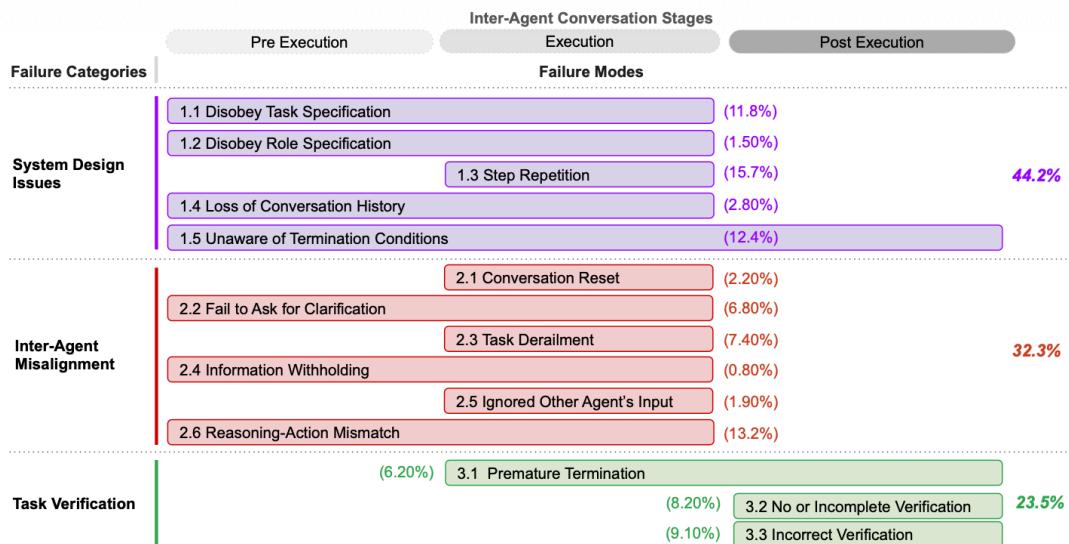


Figure 4 – Taxonomie MAST des modes d'échec des systèmes multi-agents LLM. Source : Cemri et al. (2025).

4.3 Analyse coût-efficacité et érosion de l'avantage multi-agents

Kapoor et al. (2024), dans leur article « AI Agents That Matter » (Princeton University), démontrent que l'optimisation mono-objectif (précision seule) masque des inefficacités systémiques majeures. Leur analyse révèle des variations de coût allant jusqu'à 50x entre agents atteignant des niveaux de précision similaires, et introduisent le concept de frontière de Pareto coût-précision comme méthodologie d'évaluation rigoureuse. La précision seule ne peut identifier le progrès car elle peut être améliorée par des méthodes scientifiquement dénuées de sens telles que le retry (Kapoor et al., 2024, p. 4).

Gao et al. (2025) approfondissent cette critique en comparant systématiquement systèmes single-agent et multi-agents. Leurs résultats montrent que les gains de performance MAS chutent de 10-16 % avec des modèles de capacité moyenne à seulement 0,8-3 % avec des modèles frontières, pour un surcoût en tokens de 4x à 220x (Gao et al., 2025). Les auteurs proposent des paradigmes hybrides (routing, cascade) améliorant simultanément précision (+1,1-12 %) et efficacité (-88,1 % de coût), suggérant que la question n'est pas « single-agent ou multi-agent ? » mais comment adapter dynamiquement le degré de coordination à la complexité de la tâche.

4.4 Positions industrielles et principe de parcimonie

Yan (2025), CTO de Cognition AI (créateurs de Devin), argumente dans un billet de blog technique que la fragmentation de contexte constitue le défaut architectural fondamental des systèmes multi-agents. Lorsque le contexte est distribué entre multiples agents, aucun agent individuel ne possède une vue suffisamment complète pour prendre des décisions optimales.

Anthropic (2025) recommande officiellement de commencer avec l'approche la plus simple qui fonctionne et d'ajouter de la complexité seulement quand l'évidence le supporte. Cette guidance méthodologique s'inscrit dans la tradition du rasoir d'Occam appliquée à l'ingénierie logicielle.

Cette lecture critique ne disqualifie pas la stigmergie ; elle fixe plutôt un niveau d'exigence. La suite doit donc montrer, sur des cas de migration concrets, qu'une coordination par environnement peut réellement améliorer le compromis performance-coût-gouvernabilité.

5 Migration de code et transformation avec les LLM

La migration et la transformation de code constituent le terrain d'application principal de cette recherche. Ce terrain permet de relier directement les choix d'orchestration à des contraintes industrielles réelles : qualité, coût, délai et maîtrise du risque.

5.1 Migration à grande échelle : la validation industrielle de Google

L'article de Ziftci et al. (2025), présenté au FSE 2025 (conférence top-tier en génie logiciel), constitue la première validation industrielle académiquement documentée de migration LLM-assistée à très grande échelle. Sur 39 migrations distinctes totalisant 595 changements de code et 93 574 éditions individuelles, 74,45 % des changements générés ont été appliqués sans modification humaine, réduisant de 50 % le temps développeur total requis (Ziftci et al., 2025, p. 6). L'architecture repose sur une combinaison d'analyse statique pour l'identification des emplacements, de génération LLM pour les patches, de validation automatisée via tests existants, et de review humaine sélective pour les cas complexes.

Malgré ces résultats, Ziftci et al. (2025) identifient des limitations significatives. La fenêtre de contexte limitée des LLM constraint l'efficacité pour les fichiers volumineux ou les migrations nécessitant la compréhension de multiples fichiers interdépendants (p. 11). Les tests pré-existants défaillants (*flaky tests*) empêchent la validation automatique (p. 12). Environ 25 % des changements nécessitent une intervention humaine pour des cas non anticipés.

Ces principes sont directement transposables à l'orchestration multi-agents stigmergique : les tests automatisés servent de feedback stigmergique validant la qualité des modifications, les changements peuvent être générés, validés et intégrés de manière asynchrone, et chaque changement est tracé dans le système de contrôle de version fournissant l'auditabilité requise pour la gouvernance.

5.2 Coordination multi-agents pour le codage autonome : les leçons de Cursor

L'article de Cursor (2025) documente les leçons tirées d'expérimentations avec des agents de codage autonomes fonctionnant pendant des semaines consécutives. L'approche initiale reposait sur une coordination décentralisée égalitaire où chaque agent vérifiait ce que les autres faisaient via un fichier partagé avec mécanismes de verrouillage. Cette approche a échoué de manière instructive : goulots d'étranglement des verrous (vingt agents ralentissant au débit effectif de deux ou trois), fragilité systémique (défaillances en cascade), et aversion au risque collective (sans hiérarchie, les agents évitaient les tâches difficiles) (Cursor, 2025).

Le passage à un contrôle de concurrence optimiste puis à une architecture planificateurs-travailleurs illustre une trajectoire pertinente pour la conception d'orchestrations stigmergiques. L'échec de la coordination décentralisée naïve confirme que la stigmergie ne peut se réduire à un simple fichier partagé : elle nécessite des mécanismes de sélection, de renforcement et d'évaporation inspirés des systèmes biologiques pour produire une coordination effective.

5.3 Traduction cross-language et modernisation legacy

Rozière et al. (2020), dans leur article présenté à NeurIPS 2020, démontrent la possibilité de traduction non supervisée entre langages de programmation en exploitant les mécanismes de *back-translation* et de *denoising auto-encoding*, atteignant des scores BLEU de 68,7 pour la traduction C++ vers Java et 61,9 pour Java vers C++ sans données d'entraînement parallèles (le score BLEU mesure la proximité entre une traduction générée et une traduction de référence : plus il est élevé, plus les deux sont proches). CodeRosetta (2024) étend cette approche à la programmation parallèle (NeurIPS 2024). Ces travaux établissent que la traduction cross-language par LLM est techniquement faisable, ouvrant la voie à son orchestration dans des workflows de migration multi-fichiers.

La modernisation de systèmes COBOL vers Java représente un cas industriel critique. Sneed (2010) documente les défis fondamentaux de cette migration : différences paradigmatiques entre COBOL procédural et Java orienté-objet, complexité des systèmes legacy (775 milliards de lignes en production selon IBM, 2023), et nécessité de préserver la sémantique métier exacte. Cet enjeu est central pour les secteurs bancaire, assurance et paiements, où une partie du cœur opérationnel repose encore sur du COBOL. IBM Watsonx combine analyse statique et génération LLM pour automatiser cette transformation (IBM, 2023). Diggs et al. (2025) évaluent l'utilisation de LLM pour générer la documentation de code legacy comme prérequis à la modernisation, publié dans le cadre du workshop LLM4Code@ICSE 2025.

5.4 Évolution et refactoring d'API

Lamothe, Guéhéneuc et Shang (2021) produisent dans *ACM Computing Surveys* la revue systématique la plus complète sur l'évolution des API, analysant 110 études primaires. Leur synthèse identifie les ruptures d'API comme un problème central du génie logiciel et cartographie les solutions existantes pour la migration de clients lors d'évolutions d'API. Dig et Johnson (2005) classifient les refactorings de manière à orienter les outils de migration automatisés. Ces travaux fondent les exigences fonctionnelles pour un système multi-agents de migration : identification automatique des breaking changes, génération de patches de mise à jour, et validation de la préservation sémantique.

5.5 Transformation du développeur par l'IA

Peng et al. (2023), dans une étude contrôlée randomisée, documentent une amélioration de 55,8 % de la productivité des développeurs utilisant GitHub Copilot. Barke et al. (2023) montrent de leur côté que le travail alterne entre un mode « accélération » (l'IA aide à produire plus vite) et un mode « exploration » (l'IA aide à clarifier un problème encore flou).

Ce déplacement du rôle apparaît encore plus clairement dans des retours récents de terrain. Le billet OpenAI du 5 février 2026 sur GPT-5.3-Codex décrit des équipes qui utilisent Codex pour accélérer leur propre cycle de recherche et de déploiement, avec des chercheurs et ingénieurs expliquant que leur travail est déjà « fondamentalement différent » de celui d'il y a deux mois (OpenAI, 2026). Côté Anthropic, une étude interne publiée en décembre 2025 montre aussi un glissement vers des tâches de pilotage, supervision et arbitrage, plutôt qu'une production ligne par ligne en continu (Anthropic, 2025b).

Dans ce contexte, la *pull request* (PR, proposition de changement de code soumise à revue) devient un objet clé : le développeur n'est plus seulement auteur direct du code, il devient aussi superviseur de sorties générées, arbitre de qualité et garant de cohérence.

Ces retours terrain préparent logiquement la suite : si les agents peuvent produire beaucoup plus vite, la question centrale devient la gouvernance de cette production à l'échelle.

6 Gouvernance, auditabilité et conformité

La gouvernance devient ici le point d'équilibre entre autonomie des agents et contrôle organisationnel.

6.1 Guardrails pour écologies humain-IA

Grisold, Berente et Seidel (2025) publient dans *MIS Quarterly* une contribution théorique majeure introduisant le concept d'écologies humain-IA, définies comme des environnements où agents humains et IA se coordonnent via des normes sociales plutôt que par programmation

explicite (Grisold et al., 2025, p. 1241). Leur concept central de *guardrails* est défini comme des zones de comportement désirable qui contraignent les opérations système sans éliminer l'autonomie adaptative (Grisold et al., 2025, p. 1243).

Les auteurs distinguent les normes profondes (*deep norms*), principes fondamentaux stables reflétant les valeurs organisationnelles de long terme, et les normes de surface (*surface norms*), guardrails opérationnels spécifiques pouvant évoluer selon les contextes. Leur Design Theory for Predictability s'articule autour de trois principes : l'explicitation selective des normes, la transparence des frontières comportementales, et l'évolutivité des guardrails sans reconfiguration système complète (Grisold et al., 2025, p. 1250-1253).

Cette approche possède une pertinence directe pour l'orchestration multi-agents stigmergique. Dans un système stigmergique, les guardrails peuvent être implémentés comme contraintes environnementales plutôt que comme règles programmatiques dans chaque agent. L'environnement (dépôt Git) peut n'accepter que les commits signés et liés à une issue tracker (guardrail structurel), déclencher automatiquement un workflow de revue humaine au-delà d'un seuil de complexité (guardrail de validation), et révoquer automatiquement toute modification échouant aux tests CI/CD (guardrail de rollback). Cette approche centralise la gouvernance dans l'environnement plutôt que de la distribuer dans chaque agent, facilitant l'auditabilité et l'évolution des politiques.

6.2 Perspective principal-agent et orchestration

Jarrahi et Ritala (2025), dans *California Management Review*, appliquent la théorie économique principal-agent aux agents IA, démontrant que les agents IA travaillent au nom des utilisateurs dans des contraintes définies mais disposent d'une capacité de traitement et d'accès à l'information supérieurs (Jarrahi & Ritala, 2025, p. 2). Leur framework à cinq dimensions pour la gouvernance, interactivité et adaptabilité, autonomie guidée, intelligence collective, sécurité, responsabilité et interopérabilité via orchestration, individualisation et alignement, fournit un cadre structurant pour la conception de systèmes multi-agents gouvernables (Jarrahi & Ritala, 2025).

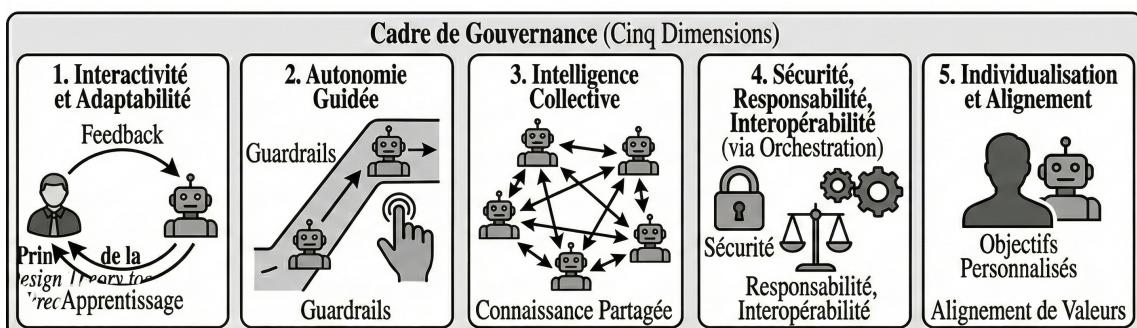


Figure 5 – Cadre de gouvernance pour systèmes multi-agents : articulation des dimensions de contrôle, responsabilité et alignement. Source : adaptation de Jarrahi et Ritala (2025).

L'emphase sur l'auditabilité et la traçabilité des décisions agents comme prérequis pour l'accountability organisationnelle (Jarrahi & Ritala, 2025, p. 9) converge avec l'approche stigmergique via systèmes de contrôle de version, où l'historique des modifications reste inspectable.

6.3 Conformité réglementaire : EU AI Act Article 14

L'Article 14 de l'EU AI Act impose que les systèmes d'IA à haut risque soient conçus et développés de manière à permettre une supervision humaine effective durant leur utilisation (cité dans Fink, 2025, p. 3). Cette formulation établit deux principes : le *design for oversight* (la supervision doit être intégrée dès la conception) et l'*effectiveness* (la supervision doit être significative, pas simplement formelle).

Holmström et al. (2023) identifient trois modèles de supervision reconnus par l'AI Act : *Human-in-the-loop* (HITL) où chaque décision requiert approbation humaine, *Human-on-the-loop* (HOTL) où le système opère de manière autonome avec surveillance et capacité d'intervention humaine, et *Human-in-command* (HIC) où les humains maintiennent un contrôle stratégique sans intervenir dans les décisions opérationnelles individuelles. Fink (2025) détaille les cinq obligations de l'Article 14(4) : compréhension des capacités et limitations, conscience du biais d'automation, interprétation correcte des outputs, capacité d'override, et capacité d'arrêt.

Pour les systèmes multi-agents stigmergiques, le mode HOTL apparaît comme le plus adapté : les agents opèrent de manière autonome via coordination indirecte, tandis que les superviseurs humains monitent les artefacts partagés et interviennent quand nécessaire. Les guardrails environnementaux (Grissold et al., 2025) incarnent le *design for oversight* en intégrant la supervision dès la conception de l'environnement partagé.

6.4 Meaningful Human Control : tracking et tracing

Santoni de Sio et van den Hoven (2018) proposent dans *Frontiers in Robotics and AI* un cadre conceptuel pour la supervision humaine effective des systèmes autonomes complexes, structuré autour de deux conditions nécessaires. La condition de *tracking* exige que le système réponde de manière appropriée aux raisons morales identifiées par les humains (Santoni de Sio & van den Hoven, 2018, p. 3), garantissant que les décisions reflètent les considérations éthiques et normatives humaines même lorsque le contrôle opérationnel direct est impossible. La condition de *tracing* exige qu'il soit possible de remonter les résultats du système à au moins un humain responsable (Santoni de Sio & van den Hoven, 2018, p. 4), garantissant l'accountability.

Pour les systèmes multi-agents stigmergiques, cette exigence est particulièrement critique car la coordination indirecte via environnement peut créer des chaînes causales opaques et les comportements émergents résultent d'interactions multiples diluant potentiellement la responsabilité. Les traces dans l'environnement (commits Git, logs) doivent permettre de reconstituer qui a configuré quelles contraintes et approuvé quelles politiques.

6.5 Dimensions organisationnelles : résistance et acceptabilité

L'introduction de systèmes multi-agents dans des workflows de développement logiciel constitue une transformation organisationnelle profonde dont la gouvernance technique ne suffit pas à garantir le succès. Markus (1983), dans un article fondateur publié dans *Communications of the ACM*, établit que les échecs d'implémentation de systèmes d'information ne résultent pas principalement de déficiences techniques, mais émergent lorsque le SI redistribue pouvoir et contrôle au sein de l'organisation (Markus, 1983, p. 430). Cette théorie interactionniste, qui attribue la résistance à l'interaction entre les caractéristiques du système et le contexte organisationnel plutôt qu'aux seules caractéristiques du système ou des utilisateurs, est particulièrement pertinente pour l'orchestration stigmergique. En effet, le passage d'une coordination hiérarchique à une coordination émergente réalloue fondamentalement trois dimensions organisationnelles : qui décide (les décisions de design peuvent basculer des développeurs seniors vers des algorithmes de coordination), qui est évalué (la performance individuelle devient visible via des métriques automatisées de contribution aux artefacts partagés), et qui contrôle le processus (la coordination peut passer de chefs de projets humains à des orchestrateurs algorithmiques). Markus (1983) conclut que la gouvernance doit inclure un travail explicite sur les enjeux de pouvoir et la relégitimation des rôles humains, une recommandation qui demeure essentielle quarante ans après sa formulation.

Kim et Kankanhalli (2009) approfondissent cette analyse dans *MIS Quarterly* en mobilisant la théorie du biais de statu quo pour modéliser la résistance des utilisateurs. Leur modèle identifie trois catégories de coûts perçus du changement, les coûts de transition (apprentissage, adaptation des routines), les coûts d'incertitude (imprévisibilité des résultats) et les coûts de submersion (*sunk costs* dans les pratiques existantes), auxquels s'ajoutent deux biais cognitifs amplificateurs : l'aversion aux pertes et l'illusion de contrôle (Kim & Kankanhalli, 2009, p. 575). Pour l'orchestration stigmergique, les coûts d'incertitude sont particulièrement saillants car le comportement émergent est, par définition, moins prévisible que le comportement d'un workflow orchestré hiérarchiquement. Kim et Kankanhalli (2009) recommandent que les systèmes soient gouvernés comme des programmes de transformation organisationnelle : réduire les coûts de basculement par des phases d'adoption graduelles, rendre la valeur tangible rapidement (*quick wins*), et prévoir des mécanismes de retour en arrière réduisant le risque perçu.

Lee et See (2004) proposent dans *Human Factors* un modèle de confiance calibrée en l'automation, qui éclaire une dimension complémentaire de la résistance. Leur cadre distingue deux pathologies de confiance : la sur-confiance (*complacency*) conduisant à une supervision insuffisante, particulièrement dangereuse lorsque les agents LLM produisent des hallucinations convaincantes dans du code, et la sous-confiance (*disuse*) conduisant au rejet de systèmes performants, ce qui annule les gains de productivité attendus. La calibration de la confiance dépend de la transparence du processus décisionnel et de la prédictibilité des résultats (Lee & See, 2004), deux propriétés que la stigmergie peut favoriser : les artefacts partagés (phéromones numé-

riques, logs de coordination) rendent le processus de coordination observable, et les patterns de dépôt/lecture de phéromones créent des régularités identifiables par les opérateurs humains.

Dietvorst, Simmons et Massey (2015) documentent dans le *Journal of Experimental Psychology : General* le phénomène d'*algorithm aversion*, la tendance systématique à rejeter les algorithmes après avoir observé même une seule erreur, tout en tolérant des taux d'erreur supérieurs chez les décideurs humains. Ce biais est particulièrement pertinent pour les systèmes multi-agents où les erreurs sont visibles (un commit incorrect, une transformation de code erronée) et attribuables au système algorithmique. Dans une étude complémentaire, Dietvorst et al. (2018) démontrent dans *Management Science* que permettre aux utilisateurs de modifier même légèrement les outputs algorithmiques, par exemple ajuster un paramètre de 2%, réduit significativement l'aversion, même lorsque cette modification n'améliore pas objectivement les résultats. Cette découverte a des implications directes pour la conception des guardrails stigmergiques : offrir aux développeurs la possibilité de paramétriser les seuils de phéromones, les critères de validation automatique et les niveaux d'autonomie des agents peut considérablement améliorer l'acceptabilité du système.

Turel et Kalhan (2023), dans *MIS Quarterly*, approfondissent encore cette analyse en montrant que l'aversion algorithmique fonctionne comme un préjugé implicite mesurable par l'Implicit Association Test (IAT), révélant que les individus associent automatiquement les algorithmes à des attributs négatifs (manque de fiabilité, froideur) même lorsque leurs jugements explicites sont neutres. Cependant, et c'est un résultat encourageant pour le déploiement organisationnel, cette aversion implicite est transitoire et diminue significativement avec l'expérience d'utilisation (Turel & Kalhan, 2023, p. 1385). Ce résultat suggère que les phases pilotes et les programmes d'exposition progressive sont des leviers efficaces de gestion du changement.

García-Ruiz et Rocchi (2025) apportent une dimension éthique complémentaire essentielle en examinant, dans *Business Ethics Quarterly*, si le travail peut rester significatif (*meaningful work*) sous management algorithmique. Leur analyse, ancrée dans la philosophie des vertus d'Alasdair MacIntyre, argumente que le travail significatif requiert l'exercice de l'agentivité morale, la capacité de prendre des décisions délibérées reflétant des valeurs et un jugement professionnel. Le management algorithmique, lorsqu'il réduit le rôle humain à celui de simple exécutant de prescriptions automatisées, peut éroder cette agentivité et transformer le travail en activité vide de sens (García-Ruiz & Rocchi, 2025). Pour l'orchestration stigmergique, cette analyse implique que la conformité à l'AI Act ne suffit pas : il faut également identifier des points de contrôle critiques où l'intervention humaine n'est pas seulement autorisée mais véritablement significative, c'est-à-dire où le jugement humain apporte une valeur que l'automatisation ne peut fournir, préservant ainsi les conditions de l'agentivité professionnelle des développeurs.

Ces six perspectives convergent vers des implications intégrées pour le déploiement d'orchestrateurs stigmergiques. L'adoption doit être progressive et gouvernée comme un programme de transformation organisationnelle (Markus, 1983 ; Kim & Kankanhalli, 2009). Les guardrails doivent être paramétrables par les utilisateurs pour réduire l'aversion algorithmique (Dietvorst et al., 2018).

La transparence des mécanismes de coordination , rendue possible par la nature même des artefacts stigmergiques, est essentielle pour maintenir une confiance calibrée (Lee & See, 2004). Des phases pilotes doivent être prévues pour capitaliser sur la transience de l'aversion implique (Turel & Kalhan, 2023). Enfin, les points de contrôle humains doivent préserver l'agentivité professionnelle et le sens du travail (García-Ruiz & Rocchi, 2025), allant au-delà d'une simple supervision formelle.

6.6 Responsabilité morale distribuée et gouvernance éthique des systèmes swarm

Floridi (2016) introduit dans *Philosophical Transactions of the Royal Society A* le concept de *Distributed Moral Actions* (DMAs) : actions moralement significatives résultant d'interactions locales individuellement neutres. Ce cadre est essentiel pour comprendre comment des comportements émergents moralement significatifs résultent d'interactions stigmergiques individuellement neutres.

Winfield et al. (2025), dans la même revue, abordent spécifiquement la gouvernance éthique des systèmes swarm en milieu réel, observant que dans les systèmes conventionnels, une propriété émergente est un bug qui doit être corrigé, mais que dans les systèmes swarm, l'émergence est une propriété désirée plutôt qu'un bug (Winfield et al., 2025). Cette distinction est fondamentale pour notre recherche : les comportements émergents des orchestrations stigmergiques ne doivent pas être supprimés mais gouvernés, ce qui nécessite des cadres de gouvernance spécifiques allant au-delà des approches traditionnelles d'assurance qualité.

Mukherjee et Chang (2025) proposent le framework d'*Operational Agency* pour tracer l'intention et la responsabilité dans les systèmes multi-agents IA, fournissant un mécanisme de reconstruction causale des décisions collectives.

6.7 Sécurité des systèmes multi-agents

Chen et al. (2024), dans un article publié à NeurIPS 2024, documentent AgentPoison, la première attaque backdoor ciblant les bases de connaissances RAG et mémoires persistantes des agents LLM, avec un taux de succès d'au moins 80 % pour un taux de poison inférieur à 0,1 % (Chen et al., 2024). L'environnement partagé, équivalent stigmergique aux phéromones numériques, devient ainsi un vecteur d'attaque persistante, créant une vulnérabilité spécifique aux architectures stigmergiques. Lee et Tiwari (2024) documentent le *Prompt Infection*, injection de prompts LLM-to-LLM se propageant silencieusement entre agents interconnectés, analogie virale particulièrement pertinente pour les systèmes de coordination indirecte.

Ces vulnérabilités imposent des exigences de sécurité spécifiques pour les orchestrations stigmergiques : authentification des traces déposées dans l'environnement, validation de l'intégrité des artefacts partagés, et mécanismes de détection d'anomalies dans les patterns de coordination

émergents.

À ce stade, la lecture conjointe des dimensions réglementaires, organisationnelles et sécurité montre qu'une orchestration stigmergique n'est viable que si la gouvernance est pensée dès le design, pas ajoutée a posteriori.

7 Agentic BPM et workflows

Le domaine émergent de l'Agentic Business Process Management (BPM) constitue un terrain d'application naturel pour l'orchestration stigmergique, à l'intersection de l'automatisation des processus et de la coordination multi-agents.

7.1 Définition formelle de l'Agentic BPM

Vu et al. (2026), dans un article présenté au BPM 2025 Responsible BPM Forum (Springer LNBIP vol. 565), établissent la première définition formelle de l'Agentic BPM basée sur une étude qualitative avec 22 praticiens. Leur conceptualisation bidirectionnelle distingue deux perspectives complémentaires : *Agents for BPM*, le déploiement d'agents logiciels autonomes pour atteindre des objectifs de processus métier, et *BPM for Agents*, l'application du BPM comme discipline pour la gouvernance des agents (Vu et al., 2026, p. 2). Cette dualité résout la tension entre autonomie et contrôle en reconnaissant que les agents nécessitent eux-mêmes une gouvernance structurée que le BPM peut fournir.

Vu et al. (2026) formulent six recommandations opérationnelles : définir des objectifs clairs plutôt que des instructions procédurales rigides (R1), établir des guardrails éthiques (R2), concevoir la collaboration humain-agent plutôt que la substitution (R3), personnaliser le comportement des agents (R4), gérer les risques proactivement (R5), et assurer une intégration sûre avec traçabilité (R6). La recommandation R1 reflète le principe stigmergique de coordination indirecte : les agents réagissent aux configurations environnementales plutôt que de suivre des scripts rigides (Heylighen, 2016a). La recommandation R6 trouve une implémentation naturelle dans les systèmes de contrôle de version identifiés par Bolici et al. (2016) comme médiums stigmergiques.

7.2 Agentic Business Process Management Systems

Dumas et al. (2026), dans un position paper présenté au AI for BPM Workshop 2025, définissent les Agentic BPMS (A-BPMS) comme une classe de systèmes d'information *process-aware* utilisant la technologie AI agentique pour exécuter des processus métier (Dumas et al., 2026, p. 2). Ils identifient trois caractéristiques distinctives : les flux d'exécution ne sont pas entièrement prédéterminés, les adaptations s'opèrent sans changements logiciels explicites, et les opportunités d'amélioration sont découvertes de manière autonome.

Dumas et al. (2026) cartographient également la trajectoire évolutive des systèmes de gestion de processus sur plusieurs décennies, depuis les systèmes papier (1960-1980) jusqu'à l'Agentic AI (2025+), en passant par les SOP, les business rules engines et le RPA. Cette progression révèle une tendance constante vers davantage d'autonomie décisionnelle, les A-BPMS représentant le point d'inflexion où les systèmes peuvent raisonner de manière véritablement autonome.

7.3 Du RPA à l'automatisation cognitive

L'évolution du RPA vers l'automatisation cognitive illustre la transition progressive vers les agents autonomes et contextualise l'émergence de l'Agentic BPM dans une trajectoire technologique plus longue. Le Cognitive RPA Framework (2025), publié dans l'*European Journal of Computer Science and Information Technology*, propose l'hybridation de l'IA avec le RPA pour créer des systèmes capables de traiter des cas non structurés, d'apprendre de l'expérience, et de s'adapter aux variations processuelles. Cette évolution s'inscrit dans la trajectoire identifiée par Dumas et al. (2026) et prépare le terrain pour les agents véritablement autonomes.

La distinction entre RPA traditionnel et automatisation cognitive est fondamentale pour comprendre la valeur ajoutée d'une approche stigmergique. Le RPA classique opère sur des règles déterministes appliquées à des interfaces utilisateur, il reproduit des séquences d'actions prédéfinies sans comprendre le contexte ni s'adapter aux variations. L'automatisation cognitive, en revanche, introduit la capacité de perception contextuelle, de raisonnement et d'apprentissage, trois propriétés que les agents LLM possèdent nativement. Le passage à l'Agentic BPM ajoute une quatrième propriété : l'autonomie décisionnelle, c'est-à-dire la capacité de l'agent à choisir ses actions en fonction de l'état de l'environnement plutôt que de suivre un script prédéterminé. C'est précisément cette propriété qui rend la coordination stigmergique pertinente : des agents autonomes nécessitent un mécanisme de coordination qui ne repose pas sur une orchestration centralisée prescrivant chaque action, mais sur un environnement partagé guidant le comportement collectif.

7.4 Systèmes adaptatifs complexes appliqués au BPM

Dooley (1997) propose un modèle de changement organisationnel basé sur les systèmes adaptatifs complexes, démontrant que les organisations opèrent comme des CAS avec des propriétés d'émergence, d'auto-organisation et de co-évolution avec leur environnement. Son cadre identifie quatre caractéristiques des CAS organisationnels, la multiplicité d'agents hétérogènes, les interactions locales non linéaires, la sensibilité aux conditions initiales et l'émergence de patterns macro à partir de comportements micro, qui s'appliquent directement aux systèmes multi-agents d'orchestration logicielle.

Uluhan et Aydin (2014) appliquent spécifiquement cette théorie au BPM, argumentant que les processus métier dans les organisations complexes se comportent comme des systèmes adaptatifs nécessitant des approches de gestion tenant compte de l'émergence et de l'auto-organisation

plutôt que d'une planification rigide et linéaire. Leur contribution principale réside dans l'identification de trois limites des approches BPM traditionnelles face à la complexité organisationnelle : l'hypothèse de prévisibilité (les processus suivent des chemins prédéterminés), l'hypothèse de décomposabilité (les processus complexes peuvent être décomposés en sous-processus indépendants), et l'hypothèse de stabilité (les processus changent lentement et de manière contrôlée). Les trois hypothèses sont violées dans les contextes de transformation de code à grande échelle, où les interdépendances entre fichiers créent des effets de cascade imprévisibles, les refactorings parallèles interfèrent mutuellement, et l'environnement technique évolue continûment (mises à jour de dépendances, changements d'API).

La convergence entre CAS et BPM stigmergique est structurelle. Dans les deux cas, la coordination émerge des interactions locales plutôt que d'une planification globale, les agents s'adaptent aux changements environnementaux sans reconfiguration centralisée, et les patterns d'activité collective émergent de manière organique plutôt que prescrite. La stigmergie fournit le mécanisme opérationnel concret qui permet aux principes abstraits des CAS de s'instancier dans des systèmes informatiques réels.

7.5 Process Mining et LLM pour le cycle de vie BPM

Berti et al. (2024) examinent comment le Process Mining peut être repensé à l'ère des agents IA, proposant que les agents puissent interagir directement avec les *event logs* pour découvrir, analyser et optimiser les processus de manière autonome. Cette vision est particulièrement pertinente pour l'orchestration stigmergique, où les traces environnementales (commits, modifications de fichiers, résultats de tests) constituent naturellement un *event log* exploitable par les techniques de Process Mining. L'intégration du Process Mining dans une architecture stigmergique permettrait de boucler la boucle d'apprentissage : les agents génèrent des traces via leurs actions sur l'environnement, le Process Mining analyse ces traces pour identifier des patterns d'efficacité ou des goulots d'étranglement, et ces analyses alimentent à leur tour les mécanismes de phéromones (renforcement des traces associées à des patterns efficaces, évaporation de celles associées à des impasses).

Vidgof, Bachhofner et Mendling (2023) identifient les opportunités et défis de l'application des LLM à l'ensemble du cycle de vie BPM, découverte de processus, analyse de conformité, identification de goulots d'étranglement, et recommandation d'améliorations, tout en soulignant les risques de hallucination et d'incohérence qui nécessitent des mécanismes de validation robustes, précisément le type de guardrails environnementaux qu'un cadre stigmergique permet d'implémenter.

Ce détour par le BPM montre qu'une coordination stigmergique peut s'inscrire dans un cadre de processus déjà connu des organisations, ce qui facilite l'appropriation côté métier et gouvernance.

8 Benchmarks et évaluation des systèmes multi-agents

L'étape suivante consiste à évaluer cette proposition avec des critères robustes, au-delà de la seule performance brute.

8.1 MultiAgentBench : évaluation de la collaboration et compétition

Zhu et al. (2025) introduisent le premier benchmark systématique pour évaluer les capacités de collaboration et de compétition dans les MAS basés sur LLM, observant que les benchmarks existants se concentrent sur les capacités d'agents individuels dans des environnements statiques, négligeant les dynamiques d'interaction (Zhu et al., 2025, p. 1). MultiAgentBench structure l'évaluation autour de scénarios collaboratifs, compétitifs et mixtes, et teste systématiquement plusieurs topologies de coordination (étoile, chaîne, arbre, graphe). La topologie impacte significativement la performance : les architectures en étoile excellent pour la centralisation de décision tandis que les graphes complets favorisent l'émergence de consensus distribués (Zhu et al., 2025, p. 8). Le benchmark introduit des métriques multidimensionnelles incluant des KPIs basés sur des jalons, des scores de complétion, et des métriques spécifiques de qualité de coordination.

8.2 REALM-Bench : planification dynamique

Geng et Chang (2025) introduisent REALM-Bench, évaluant les MAS dans des contextes de planification et scheduling sous contraintes dynamiques. Identifiant une lacune critique, la plupart des benchmarks évaluent dans des environnements statiques alors que les déploiements réels impliquent perturbations continues (Geng & Chang, 2025, p. 2), REALM-Bench intègre perturbations dynamiques, contraintes de ressources évolutives et nécessité de réoptimisation continue. Son intégration prête à l'emploi avec cinq frameworks majeurs (LangGraph, OpenAI Swarm, CrewAI, AutoGen, ALAS) permet des comparaisons directes standardisées (Geng & Chang, 2025, p. 9).

8.3 Métriques pour la génération de code

Ghosh Paul et al. (2024) produisent la synthèse la plus complète des métriques d'évaluation pour la génération de code par LLM, classifiant les métriques en quatre catégories : similarité textuelle (BLEU, ROUGE, CodeBLEU), correction fonctionnelle (Pass@k), qualité logicielle (lisibilité, maintenabilité, sécurité), et métriques sémantiques évaluant la préservation de la signification. Le benchmark RACE (2025) propose d'aller au-delà de la correction pour évaluer la lisibilité, l'efficacité, la maintenabilité et la documentation du code généré.

SWE-bench (Jimenez et al., 2024) s'est imposé comme référence pour l'évaluation des agents de génie logiciel, structuré autour d'issues GitHub réelles nécessitant compréhension du contexte,

localisation de bugs et génération de patches. SWE-bench Pro (Scale AI, 2025) étend cette évaluation avec des problèmes plus complexes nécessitant modifications multi-fichiers.

8.4 Coût-efficacité et reproductibilité

Kapoor et al. (2024) établissent la nécessité de frontières de Pareto coût-précision comme méthodologie d'évaluation, argumentant que les agents qui importent sont ceux qui offrent le meilleur compromis coût-performance plutôt que la précision maximale à tout prix. Desai et al. (2025) examinent la reproductibilité de la recherche sur l'IA, proposant des métriques distributionnelles plutôt que déterministes pour les systèmes intrinsèquement stochastiques.

Avec ces repères d'évaluation, le passage au cadre conceptuel devient plus solide : on ne propose pas seulement une idée d'architecture, on prépare aussi la manière de la tester de façon crédible.

9 Cadre conceptuel et identification du gap

9.1 Synthèse croisée de la littérature

L'analyse des sept blocs thématiques révèle un paysage de recherche riche mais fragmenté. La stigmergie est bien théorisée comme mécanisme de coordination universel (Heylighen, 2016a, 2016b) et formalisée pour les agents cognitifs (Ricci et al., 2007). Un mouvement de convergence se dessine d'ailleurs : des frameworks récents inspirés de l'intelligence en essaim, ainsi que des sorties industrielles (dont Kimi 2.5), adoptent progressivement cette logique stigmergique. Les systèmes multi-agents LLM font l'objet d'une recherche active, mais privilégient encore majoritairement les architectures hiérarchiques dont les limitations sont documentées (Cemri et al., 2025 ; Gao et al., 2025 ; Kapoor et al., 2024). Autrement dit, les briques techniques existent désormais en grande partie, et le défi se déplace vers l'intégration et l'orchestration cohérente de ces éléments. La migration de code avec LLM est validée industriellement (Ziftci et al., 2025) mais manque de cadres d'orchestration multi-agents. La gouvernance de l'IA est traitée au niveau réglementaire (Fink, 2025) et organisationnel (Grisold et al., 2025 ; Jarrahi & Ritala, 2025) mais sans considérer les spécificités des systèmes à coordination émergente.

9.2 Gap identifié

Le gap principal réside dans l'absence d'un cadre intégrateur combinant :

1. Les principes de coordination stigmergique (Heylighen, 2016a ; Ricci et al., 2007) appliqués aux agents LLM plutôt qu'aux agents réactifs simples.
2. Les exigences de gouvernance et auditabilité (Grisold et al., 2025 ; Fink, 2025 ; Santoni de Sio & van den Hoven, 2018) opérationnalisées pour des systèmes à coordination émergente.

3. La validation empirique sur des tâches de transformation de code à grande échelle (Ziftci et al., 2025), avec des métriques multidimensionnelles (Kapoor et al., 2024) allant au-delà de la seule précision.

Aucune étude existante ne propose, à notre connaissance, un système d'orchestration multi-agents LLM à coordination stigmergique évalué empiriquement sur des tâches de migration de code avec des métriques intégrant performance, coût, gouvernabilité et conformité réglementaire.

9.3 Relations conceptuelles entre les blocs thématiques

L'articulation entre les sept blocs thématiques de cette revue révèle un réseau de relations conceptuelles structurant la contribution de ce mémoire. Les fondements théoriques de la stigmergie fournissent les principes de décentralisation, construction incrémentale et flexibilité adaptative, ensuite opérationnalisés dans les mécanismes computationnels de coordination par artefacts. Les contre-arguments (Agentless, MAST, coût-efficacité) contraignent le design en fixant les baselines à dépasser et les risques à éviter. Les cas de migration (Google, Cursor, COBOL, évolution d'API) définissent le domaine applicatif. Les analyses de gouvernance (guardrails, principal-agent, AI Act, Meaningful Human Control) apportent les contraintes de traçabilité, conformité et acceptabilité. L'Agentic BPM fournit le cadre processuel intégrateur, et les benchmarks (MultiAgentBench, REALM-Bench, RACE) structurent la méthode d'évaluation.

Cette structure révèle que le gap identifié n'est pas simplement une absence de travail sur un sujet donné, mais une absence d'intégration entre des champs de recherche riches mais cloisonnés. La contribution originale de ce mémoire réside précisément dans cette intégration, proposant un cadre uniifié où les mécanismes stigmergiques servent simultanément de support de coordination technique, de vecteur de traçabilité pour la gouvernance, et d'infrastructure d'évaluation pour la validation empirique.

9.4 Complexity Leadership Theory et transformation du rôle managérial

L'adoption d'orchestrations stigmergiques dans les organisations implique une transformation profonde du rôle managérial. Uhl-Bien, Marion et McKelvey (2007), dans un article fondamental publié dans *The Leadership Quarterly*, proposent la Complexity Leadership Theory (CLT) comme cadre pour comprendre le passage du leadership bureaucratique industriel au leadership adaptatif de l'ère de la connaissance. La CLT distingue trois fonctions de leadership interdépendantes : le leadership administratif (structures, planification, allocation de ressources), le leadership adaptatif (émergence créative, apprentissage organisationnel, innovation), et le leadership enabling (création des conditions permettant l'émergence adaptive) (Uhl-Bien et al., 2007).

Cette théorie s'applique directement à la gouvernance des systèmes stigmergiques : le leadership administratif définit les guardrails et contraintes environnementales (Grisold et al., 2025),

le leadership adaptatif reconnaît et exploite les comportements émergents plutôt que de les supprimer, et le leadership enabling crée les conditions (infrastructure technique, culture organisationnelle, formation) permettant la coordination stigmergique efficace. La CLT légitime ainsi théoriquement le passage d'un management directif des workflows (orchestration hiérarchique) à un management facilitateur de l'auto-organisation (orchestration stigmergique).

Shrestha, Ben-Menahem et von Krogh (2019) complètent cette perspective dans *California Management Review* en proposant un framework des structures décisionnelles humain-IA : délégation complète à l'IA, décision séquentielle hybride, et décision agrégée humain-IA. Ces trois structures décisionnelles correspondent à des niveaux d'autonomie agentique croissants et doivent être sélectionnées en fonction de cinq facteurs contingents incluant la prévisibilité de la tâche, la disponibilité des données et les enjeux éthiques (Shrestha et al., 2019). Pour l'orchestration stigmergique de la migration de code, la décision séquentielle hybride semble la plus adaptée : les agents génèrent et proposent des transformations (première phase), les guardrails environnementaux filtrent et valident automatiquement (deuxième phase), et les humains interviennent pour les cas complexes ou critiques (troisième phase).

9.5 Cadre conceptuel proposé

Le cadre conceptuel de ce mémoire repose sur trois piliers articulant les contributions de la littérature :

Pilier conceptuel, Stigmergie cognitive et coordination par artefacts. La stigmergie cognitive (Ricci et al., 2007) fournit le cadre théorique pour des agents LLM rationnels se coordonnant via des artefacts numériques manipulables. Le modèle de coordination de Heylighen (2016a, 2016b) guide la conception des mécanismes de dépôt, évaporation et renforcement des traces. Les espaces de tuples de Gelernter (1985) et leur extension distribuée TOTA (Mamei & Zambonelli, 2009) fournissent l'infrastructure computationnelle.

Pilier managérial, Gouvernance et acceptabilité. Les guardrails de Grisold et al. (2025) structurent les contraintes environnementales. La perspective principal-agent de Jarrahi et Ritala (2025) cadre la relation utilisateur-agents. La Complexity Leadership Theory de Uhl-Bien et al. (2007) théorise le passage du command-and-control à l'orchestration d'éco- logies d'agents. Les structures décisionnelles de Shrestha et al. (2019) définissent les modes de collaboration humain-IA.

Pilier technique, Conformité et évaluation. Le Meaningful Human Control (Santoni de Sio & van den Hoven, 2018) opérationnalise les exigences de l'Article 14 de l'EU AI Act (Fink, 2025). Les frontières de Pareto coût-précision de Kapoor et al. (2024) structurent l'évaluation. Le baseline Agentless (Xia et al., 2024) et le diagnostic MAST (Cemri et al., 2025) fournissent les références comparatives.

10 Design de recherche

10.1 Approche méthodologique

Cette recherche adopte une démarche exploratoire mixte, combinant une dimension expérimentale (proof-of-concept) et une dimension qualitative (entretiens semi-directifs). L'approche exploratoire est justifiée par la nouveauté du sujet, aucun système d'orchestration multi-agents LLM à coordination stigmergique n'ayant été documenté dans la littérature académique, et par la nécessité de comprendre les mécanismes de coordination émergente avant de pouvoir les optimiser.

Le positionnement épistémologique relève du pragmatisme méthodologique (Creswell & Creswell, 2018), qui légitime l'intégration de méthodes quantitatives et qualitatives au service d'une même question de recherche. Ce choix se distingue du positivisme pur (qui imposerait une démarche exclusivement expérimentale insuffisante pour capturer les dimensions managériales et organisationnelles) et de l'interprétivisme pur (qui ne permettrait pas de valider la faisabilité technique du mécanisme proposé). L'approche mixte permet de répondre simultanément aux questions techniques (RQ1, RQ2) via le POC et aux questions managériales (RQ3) via les entretiens, tout en assurant une triangulation méthodologique renforçant la validité interne des résultats.

10.2 Champ d'étude

Le champ d'étude se situe à l'intersection de trois domaines : le génie logiciel (migration et transformation de code), les systèmes multi-agents (orchestration et coordination), et le management des systèmes d'information (gouvernance et conformité). Le terrain empirique est constitué de dépôts de code open source permettant l'évaluation reproductible de migrations automatisées. Ce choix permet de construire un prototype opérationnel dans un cadre contrôlé, puis de mesurer sa performance sur des cas réalistes. La construction du dispositif sera également informée par des entretiens d'experts afin d'aligner l'architecture proposée avec les contraintes organisationnelles et professionnelles observées sur le terrain. Dans un second temps, l'approche est pensée pour un test en grandeur nature, d'abord dans un cadre de laboratoire au sein de BNP Personal Finance, puis dans d'autres organisations de type PME/startup, sous réserve des conditions d'accès et de conformité. Ce positionnement interdisciplinaire est cohérent avec la formation de l'EMLV qui vise à former des managers comprenant les enjeux technologiques et capables de gouverner l'innovation.

10.3 Collecte de données

La collecte de données s'appuie sur deux sources complémentaires.

Source 1, Proof-of-concept (POC). Un prototype d'orchestration stigmergique sera développé et évalué sur des tâches de migration de code (par exemple, Python 2 vers Python 3 ou JavaScript vers TypeScript). L'architecture du POC s'inspirera directement du cadre conceptuel présenté dans cette revue : un environnement stigmergique central (dépôt de code + métadonnées de coordination), des agents LLM spécialisés (analyseur, transformateur, testeur, validateur) interagissant uniquement via l'environnement partagé, et des guardrails environnementaux implémentant les contraintes de gouvernance. Le POC générera des données quantitatives (taux de succès par fichier et par projet, coût en tokens par transformation, temps d'exécution total, taux de rollback, couverture de tests avant/après) et des traces d'exécution qualitatives (séquences d'actions des agents, patterns de lecture/écriture des phéromones, cas d'émergence ou d'impasse). Ces traces permettront l'analyse des mécanismes stigmergiques *in vivo*, en identifiant comment les phéromones numériques guident effectivement le comportement collectif des agents.

Source 2, Entretiens semi-directifs. Un minimum de 10 entretiens semi-directifs seront conduits avec des praticiens relevant de deux profils complémentaires : des ingénieurs logiciels expérimentés dans la migration de code et/ou l'utilisation d'outils d'IA pour le développement (profil technique, ~5 entretiens), et des managers SI, responsables de la qualité logicielle ou responsables conformité impliqués dans la gouvernance de projets d'automatisation (profil managérial, ~5 entretiens). Le guide d'entretien sera structuré autour de trois thèmes : les pratiques actuelles de migration et de coordination (contexte), la réaction au modèle stigmergique présenté via une démonstration du POC (évaluation), et les exigences de gouvernance, traçabilité et acceptabilité perçues (gouvernance). Les entretiens seront enregistrés et transcrits intégralement pour l'analyse.

10.4 Justification des choix méthodologiques

Le choix d'une approche mixte (POC + entretiens) est justifié par la triple dimension de la problématique. La dimension technique (RQ1) nécessite le développement d'un prototype fonctionnel pour identifier empiriquement les types de phéromones numériques et les règles locales adaptés. La dimension performance (RQ2) nécessite des données quantitatives comparatives obtenues via le POC. La dimension managériale (RQ3) nécessite une compréhension des perceptions et exigences des praticiens qui ne peut être captée que par des entretiens approfondis. Cette triangulation méthodologique croise données expérimentales et données perceptuelles, renforçant la validité des conclusions.

Le scénario de migration envisagé (Python 2 vers Python 3 ou JavaScript vers TypeScript) constitue une priorité opérationnelle selon trois critères. Premièrement, la faisabilité : les règles de migration sont bien documentées, les dépôts open source sont disponibles en abondance, et les outils de test automatisé existent pour valider les transformations. Deuxièmement, l'évaluabilité : des métriques claires et non ambiguës permettent de juger la qualité des résultats (tests qui passent, *type coverage*, absence de régressions). Troisièmement, la pertinence industrielle : ces

migrations correspondent à des cas d'usage réels et récurrents dans les organisations (fin de support Python 2 en janvier 2020, adoption croissante de TypeScript dans l'écosystème JavaScript). Cette priorisation n'exclut pas l'évaluation du POC sur d'autres workflows open source et benchmarks de référence, notamment TravelPlanner, afin de tester la robustesse du cadre au-delà du seul périmètre de migration.

10.5 Analyse des données

Les données quantitatives du POC seront analysées selon plusieurs cadres complémentaires. Les frontières de Pareto coût-précision (Kapoor et al., 2024) permettront de situer l'approche stigmergique par rapport à l'ensemble des alternatives existantes sur le plan de l'efficience économique. La comparaison au baseline Agentless (Xia et al., 2024) établira si la complexité additionnelle de la coordination multi-agents est justifiée par des gains de performance mesurables. Les métriques multidimensionnelles discutées dans la section benchmarks (MultiAgentBench, REALM-Bench, RACE) fourniront un cadre d'évaluation structuré couvrant correction fonctionnelle, efficacité des ressources, robustesse et adaptabilité.

Les données qualitatives des entretiens seront analysées par codage thématique en trois cycles (Miles, Huberman & Saldaña, 2020) : un codage ouvert identifiant les catégories émergentes, un codage axial structurant les relations entre catégories, et un codage sélectif intégrant les résultats dans le cadre conceptuel. La taxonomie MAST des 14 modes d'échec (Cemri et al., 2025) servira de grille de lecture pour les risques techniques identifiés par les praticiens. Les recommandations de Vu et al. (2026) structureront l'analyse des exigences de gouvernance. Le framework principal-agent de Jarrahi et Ritala (2025) cadrera l'analyse des enjeux de confiance et de contrôle.

10.6 Limites anticipées

Plusieurs limites méthodologiques sont anticipées. Le POC évaluera un ensemble de cas prioritairement orientés migration, complétés par des benchmarks ouverts comme TravelPlanner. Malgré cette ouverture, la généralisabilité restera conditionnée par le volume de cas effectivement exécutés et la diversité des dépôts retenus. L'échantillon d'entretiens (minimum 10) vise la saturation thématique mais ne prétend pas à la représentativité statistique. Les résultats du POC dépendront des modèles LLM disponibles au moment du développement, dont les capacités évoluent rapidement. Le protocole expérimental peut également engendrer des coûts d'API non négligeables (inférence, itérations de correction, exécution répétée des évaluations), ce qui peut contraindre l'ampleur des campagnes de test et la comparabilité des résultats selon le budget mobilisable. Enfin, le contexte mono-chercheur du mémoire limite les possibilités de triangulation par les investigateurs. Ces limites sont reconnues et seront discutées dans l'analyse des résultats.

11 Bibliographie

- Amazon. (2024). *Amazon Q Developer Agent : Java migration capabilities*. <https://aws.amazon.com/q/developer/>
- Anthropic. (2025). *When to use multi-agent systems (and when not to)*. Claude Documentation. <https://docs.anthropic.com/>
- Anthropic. (2025b, December 2). *How AI is transforming work at Anthropic*. <https://www.anthropic.com/news/how-ai-is-transforming-work-at-anthropic>
- Anthropic. (2025c). *Model Context Protocol (MCP)*. <https://modelcontextprotocol.io/>
- Anthropic. (2026). *Claude Code*. <https://claude.ai/code>
- Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded Copilot : How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7(OOPSLA1), 85-111. <https://doi.org/10.1145/3586030>
- Berti, A., Maatallah, M., Jessen, U., Sroka, M., & Ghannouchi, S. A. (2024). Re-thinking process mining in the AI-based agents era. *arXiv preprint arXiv:2408.07720*. <https://doi.org/10.48550/arXiv.2408.07720>
- Bolici, F., Howison, J., & Crowston, K. (2016). Stigmergic coordination in FLOSS development teams : Integrating explicit and implicit mechanisms. *Cognitive Systems Research*, 38, 14-22. <https://doi.org/10.1016/j.cogsys.2015.12.003>
- Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence : From natural to artificial systems*. Oxford University Press.
- Carriero, N., & Gelernter, D. (1992). Coordination languages and their significance. *Communications of the ACM*, 35(2), 97-107. <https://doi.org/10.1145/129630.129635>
- Cemri, M., Pan, M. Z., Yang, S., et al. (2025). Why do multi-agent LLM systems fail ? *arXiv preprint arXiv:2503.13657*. <https://doi.org/10.48550/arXiv.2503.13657>
- Chari, A., Tiwari, A., Lian, R., Reddy, S., & Zhou, B. (2025). Pheromone-based learning of optimal reasoning paths. *arXiv preprint arXiv:2501.19278*. <https://doi.org/10.48550/arXiv.2501.19278>
- Chen, Z., Zhao, Z., Zhou, A., Huang, Y., Wang, D., Poon, J., & Li, J. (2024). AgentPoison : Red-teaming LLM agents via poisoning memory or knowledge bases. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*. <https://arxiv.org/abs/2407.12784>
- CodeRosetta. (2024). Pushing the boundaries of unsupervised code translation for parallel programming. *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.
- Cognitive RPA Framework. (2025). Cognitive RPA : A framework for hybridizing artificial intelligence with robotic process automation. *European Journal of Computer Science and Information Technology*, 13(19), 64-78.
- Creswell, J. W., & Creswell, J. D. (2018). *Research design : Qualitative, quantitative, and mixed methods approaches* (5th ed.). SAGE Publications.

- Cursor. (2025). Scaling long-running autonomous coding. *Cursor Blog*. <https://cursor.com/blog/scaling-agents>
- Desai, A., et al. (2025). What is reproducibility in artificial intelligence and machine learning research? *arXiv preprint arXiv:2407.10239*. <https://doi.org/10.48550/arXiv.2407.10239>
- Dietvorst, B. J., Simmons, J. P., & Massey, C. (2015). Algorithm aversion : People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology : General*, 144(1), 114-126. <https://doi.org/10.1037/xge0000033>
- Dietvorst, B. J., Simmons, J. P., & Massey, C. (2018). Overcoming algorithm aversion : People will use algorithms if they can (even slightly) modify them. *Management Science*, 64(3), 1155-1170. <https://doi.org/10.1287/mnsc.2016.2643>
- Dig, D., & Johnson, R. (2005). The role of refactorings in API evolution. In *IEEE International Conference on Software Maintenance (ICSM 2005)* (pp. 2-11). <https://doi.org/10.1109/ICSM.2005.26>
- Diggs, C., Doyle, M., Madan, A., & Scott, E. O. (2025). Leveraging LLMs for legacy code modernization : Evaluation of LLM-generated documentation. In *2025 IEEE/ACM International Workshop on Large Language Models for Code (LLM4Code@ICSE 2025)*. <https://doi.org/10.1109/LLM4Code66737.2025.00027>
- Dooley, K. (1997). A complex adaptive systems model of organization change. *Nonlinear Dynamics, Psychology, and Life Sciences*, 1(1), 69-97. <https://doi.org/10.1023/A:1022375910940>
- Dumas, M., Leopold, H., Mendling, J., Rinderle-Ma, S., Smedt, J. D., & Zimoch, M. (2026). Agentic business process management systems. *arXiv preprint arXiv:2601.18833*. <https://doi.org/10.48550/arXiv.2601.18833>
- Fink, M. (2025). Human oversight under Article 14 of the EU AI Act. *SSRN*. <https://ssrn.com/abstract=5147196>
- Floridi, L. (2016). Faultless responsibility : On the nature and allocation of moral responsibility for distributed moral actions. *Philosophical Transactions of the Royal Society A*, 374(2083), 20160112. <https://doi.org/10.1098/rsta.2016.0112>
- Gao, M., Li, Y., Liu, B., et al. (2025). Single-agent or multi-agent systems ? Why not both ? *arXiv preprint arXiv:2505.18286*. <https://doi.org/10.48550/arXiv.2505.18286>
- García-Ruiz, P., & Rocchi, M. (2025). Can work remain meaningful under algorithmic management? A MacIntyrean analysis. *Business Ethics Quarterly*, 35(1), 1-28. <https://doi.org/10.1017/beq.2024.36>
- Gelernter, D. (1985). Generative communication in Linda. *ACM Transactions on Programming Languages and Systems*, 7(1), 80-112. <https://doi.org/10.1145/2363.2433>
- Geng, L., & Chang, E. Y. (2025). REALM-Bench : A benchmark for evaluating multi-agent systems on real-world, dynamic planning and scheduling tasks. *arXiv preprint arXiv:2502.18836*. <https://doi.org/10.48550/arXiv.2502.18836>
- Ghosh Paul, D., Zhu, H., & Bayley, I. (2024). Benchmarks and metrics for evaluations of code generation : A critical review. *arXiv preprint arXiv:2406.12655*. <https://doi.org/10.48550/arXiv.2406.12655>

48550/arXiv.2406.12655

Grassé, P.-P. (1959). La reconstruction du nid et les coordinations interindividuelles chez *Bellicositermes natalensis* et *Cubitermes* sp. la théorie de la stigmergie. *Insectes Sociaux*, 6(1), 41-80. <https://doi.org/10.1007/BF02223791>

Grisold, T., Berente, N., & Seidel, S. (2025). Guardrails for human-AI ecologies : Norm-based coordination and design for predictability. *MIS Quarterly*, 49(4), 1239-1266. <https://doi.org/10.25300/MISQ/2025/18058>

Heylighen, F. (2016a). Stigmergy as a universal coordination mechanism I : Definition and components. *Cognitive Systems Research*, 38, 4-13. <https://doi.org/10.1016/j.cogsys.2015.12.002>

Heylighen, F. (2016b). Stigmergy as a universal coordination mechanism II : Varieties and evolution. *Cognitive Systems Research*, 38, 50-59. <https://doi.org/10.1016/j.cogsys.2015.12.007>

Holland, J. H. (1995). *Hidden order : How adaptation builds complexity*. Addison-Wesley.

Holmström, J., et al. (2023). 'Human oversight' in the EU artificial intelligence act : What, when and by whom ? *Law, Innovation and Technology* (Taylor & Francis). <https://doi.org/10.1080/17579961.2023.2293129>

Hong, S., Zhuge, M., Chen, J., et al. (2024). MetaGPT : Meta programming for a multi-agent collaborative framework. *arXiv preprint arXiv :2308.00352*. <https://doi.org/10.48550/arXiv.2308.00352>

IBM. (2023). *IBM unveils watsonx generative AI capabilities to accelerate mainframe application modernization*. <https://newsroom.ibm.com/2023-08-22-IBM-Unveils-watsonx-Generative-AI-Capabilities-to-Accelerate>Mainframe-Application-Modernization>

Jarrahi, M. H., & Ritala, P. (2025, July 23). Rethinking AI agents : A principal-agent perspective. *California Management Review* (Insight). <https://cmr.berkeley.edu/2025/07/rethinking-ai-agents-a-principal-agent-perspective/>

Jimenez, C. E., Yang, J., Wettig, A., et al. (2024). SWE-bench : Can language models resolve real-world GitHub issues ? *arXiv preprint arXiv :2310.06770*. <https://doi.org/10.48550/arXiv.2310.06770>

Kapoor, S., Stroebel, B., Siegel, Z. S., Nadgir, N., & Narayanan, A. (2024). AI agents that matter. *arXiv preprint arXiv :2407.01502*. <https://doi.org/10.48550/arXiv.2407.01502>

Kauffman, S. A. (1993). *The origins of order : Self-organization and selection in evolution*. Oxford University Press.

Kim, H.-W., & Kankanhalli, A. (2009). Investigating user resistance to information systems implementation : A status quo bias perspective. *MIS Quarterly*, 33(3), 567-582. <https://doi.org/10.2307/20650309>

Lamothe, M., Guéhéneuc, Y.-G., & Shang, W. (2021). A systematic review of API evolution literature. *ACM Computing Surveys*, 54(8), 171 :1-171 :36. <https://doi.org/10.1145/3450268>

Lee, D., & Tiwari, M. (2024). Prompt injection : LLM-to-LLM prompt injection within

- multi-agent systems. *arXiv preprint arXiv* :2410.07283. <https://arxiv.org/abs/2410.07283>
- Lee, J. D., & See, K. A. (2004). Trust in automation : Designing for appropriate reliance. *Human Factors*, 46(1), 50-80. https://doi.org/10.1518/hfes.46.1.50_30392
- Liu, X., Yu, H., Zhang, H., et al. (2024). AgentBench : Evaluating LLMs as agents. *arXiv preprint arXiv* :2308.03688. <https://doi.org/10.48550/arXiv.2308.03688>
- Mamei, M., & Zambonelli, F. (2009). Programming pervasive and mobile computing applications : The TOTA approach. *ACM Transactions on Software Engineering and Methodology*, 18(4), 1-56. <https://doi.org/10.1145/1538942.1538945>
- Markus, M. L. (1983). Power, politics, and MIS implementation. *Communications of the ACM*, 26(6), 430-444. <https://doi.org/10.1145/358141.358148>
- Marsh, L., & Onof, C. (2008). Stigmergic epistemology, stigmergic cognition. *Cognitive Systems Research*, 9(1-2), 136-149. <https://doi.org/10.1016/j.cogsys.2007.06.009>
- Miles, M. B., Huberman, A. M., & Saldaña, J. (2020). *Qualitative data analysis : A methods sourcebook* (4th ed.). SAGE Publications.
- Mukherjee, A., & Chang, H. (2025). Operational agency : A framework for tracing intent and liability in multi-agent artificial intelligence systems. *SSRN Electronic Journal*. <https://doi.org/10.2139/ssrn.5136065>
- Parunak, H. V. D. (1997). "Go to the ant" : Engineering principles from natural agent systems. *Annals of Operations Research*, 75, 69-101. <https://doi.org/10.1023/A:1018980001403>
- OpenAI. (2026, February 5). *Introducing GPT-5.3-Codex*. <https://openai.com/index/introducing-gpt-5-3-codex/>
- OpenAI. (2022, November 30). *Introducing ChatGPT*. <https://openai.com/index/chatgpt/>
- OpenAI. (2026b). *Codex CLI Documentation*. <https://developers.openai.com/codex/cli>
- Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity : Evidence from GitHub Copilot. *arXiv preprint arXiv* :2302.06590. <https://doi.org/10.48550/arXiv.2302.06590>
- Rahman, M. A. U., Schranz, M., & Hayat, S. (2025). LLM-powered swarms : A new frontier or a conceptual stretch ? *arXiv preprint arXiv* :2506.14496. <https://doi.org/10.48550/arXiv.2506.14496>
- Ricci, A., Omicini, A., Viroli, M., Gardelli, L., & Oliva, E. (2007). Cognitive stigmergy : Towards a framework based on agents and artifacts. In D. Weyns, H. V. D. Parunak, & F. Michel (Eds.), *Environments for Multi-Agent Systems III* (LNCS Vol. 4389, pp. 124-140). Springer. https://doi.org/10.1007/978-3-540-71103-2_7
- Rodriguez, R. R., Jr. (2026). Emergent coordination in multi-agent systems via pressure fields and temporal decay. *arXiv preprint arXiv* :2601.08129. <https://doi.org/10.48550/arXiv.2601.08129>
- Rozière, B., Lachaux, M.-A., Chanussot, L., & Lample, G. (2020). Unsupervised translation of programming languages. *Advances in Neural Information Processing Systems 33 (NeurIPS*

2020), 20601-20611. <https://doi.org/10.48550/arXiv.2009.08732>

Santoni de Sio, F., & van den Hoven, J. (2018). Meaningful human control over autonomous systems : A philosophical account. *Frontiers in Robotics and AI*, 5, Article 15. <https://doi.org/10.3389/frobt.2018.00015>

Serugendo, G. D. M., Gleizes, M. P., & Karageorgos, A. (2005). Self-organization in multi-agent systems. *The Knowledge Engineering Review*, 20(2), 165-189. <https://doi.org/10.1017/S0269888905000494>

Shrestha, Y. R., Ben-Menahem, S. M., & von Krogh, G. (2019). Organizational decision-making structures in the age of artificial intelligence. *California Management Review*, 61(4), 66-83. <https://doi.org/10.1177/0008125619862257>

Sneed, H. M. (2010). Migrating from COBOL to Java. In *IEEE International Conference on Software Maintenance (ICSM 2010)* (pp. 237-246). <https://doi.org/10.1109/ICSM.2010.5609741>

Theraulaz, G., & Bonabeau, E. (1999). A brief history of stigmergy. *Artificial Life*, 5(2), 97-116. <https://doi.org/10.1162/106454699568700>

Turel, O., & Kalhan, S. (2023). Prejudiced against the machine ? Implicit associations and the transience of algorithm aversion. *MIS Quarterly*, 47(4), 1369-1394. <https://doi.org/10.25300/MISQ/2023/16860>

Uhl-Bien, M., Marion, R., & McKelvey, B. (2007). Complexity leadership theory : Shifting leadership from the industrial age to the knowledge era. *The Leadership Quarterly*, 18(4), 298-318. <https://doi.org/10.1016/j.leaqua.2007.04.002>

Uluhan, E., & Aydin, M. N. (2014). Complex adaptive systems theory in the context of business process management. In *S-BPM ONE 2014 : Scientific Research* (pp. 11-23). Springer. https://doi.org/10.1007/978-3-319-06191-7_2

Vidgof, M., Bachhofner, S., & Mendling, J. (2023). Large language models for business process management : Opportunities and challenges. *arXiv preprint arXiv:2304.04309*. <https://doi.org/10.48550/arXiv.2304.04309>

Vu, H., Klievtsova, N., Leopold, H., Rinderle-Ma, S., & Kampik, T. (2026). Agentic business process management : Practitioner perspectives on agent governance in business processes. In *BPM 2025 Responsible BPM Forum. Lecture Notes in Business Information Processing* (vol. 565). Springer. <https://doi.org/10.48550/arXiv.2504.03693>

Wang, X., Li, B., Song, Y., et al. (2025). OpenHands : An open platform for AI software developers as generalist agents. *arXiv preprint arXiv:2407.16741*. <https://doi.org/10.48550/arXiv.2407.16741>

Weyns, D., Omicini, A., & Odell, J. (2007). Environment as a first class abstraction in multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 14(1), 5-30. <https://doi.org/10.1007/s10458-006-0012-0>

Winfield, A. F. T., et al. (2025). On the ethical governance of swarm robotic systems in the real world. *Philosophical Transactions of the Royal Society A*, 383(2289), 20240142. <https://doi.org/10.1098/rsta.20240142>

//doi.org/10.1098/rsta.2024.0142

Xia, C. S., Deng, Y., Dunn, S., & Zhang, L. (2024). Agentless : Demystifying LLM-based software engineering agents. *arXiv preprint arXiv:2407.01489*. <https://doi.org/10.48550/arXiv.2407.01489>

Yan, W. (2025, June 12). Don't build multi-agents. *Cognition AI Blog*. <https://www.cognition-labs.com/blog>

Yang, J., Jimenez, C. E., Yao, S., et al. (2024). SWE-agent : Agent-computer interfaces enable automated software engineering. *arXiv preprint arXiv:2405.15793*. <https://doi.org/10.48550/arXiv.2405.15793>

Zhang, Y., Lin, C., Tang, S., Chen, H., Zhou, S., Ma, Y., & Tresp, V. (2025). SwarmAgentic : Towards fully automated agentic system generation via swarm intelligence. *arXiv preprint arXiv:2506.15672*. <https://doi.org/10.48550/arXiv.2506.15672>

Zhu, K., Du, H., Hong, Z., et al. (2025). MultiAgentBench : Evaluating the collaboration and competition of LLM agents. *arXiv preprint arXiv:2503.01935*. <https://doi.org/10.48550/arXiv.2503.01935>

Ziftci, C., Zheng, J., Choudhary, S., Rossbach, C., Chen, W., Li, W., Wang, J., & Zhang, L. (2025). Migrating code at scale with LLMs at Google. In *Proceedings of the ACM International Conference on the Foundations of Software Engineering (FSE 2025)*.

A Annexe IA : usage, comparaison et traçabilité

A.1 Pourquoi et comment l'IA a été utilisée

L'usage de l'IA dans ce mémoire a été cadré comme un support à la recherche et à la rédaction, pas comme un substitut au jugement académique. L'objectif opérationnel était triple : accélérer la phase exploratoire, fiabiliser la recherche documentaire via un assistant outillé sur corpus local, et améliorer la qualité formelle du document LaTeX.

Les usages effectivement retenus ont concerné :

- l'exploration et la structuration initiale de la littérature ;
- l'interrogation d'un corpus local via MCP pour obtenir des réponses ancrées dans des passages sources ;
- l'assistance technique et éditoriale pour la production du document LaTeX.

A.2 Ce qui a été conservé, modifié, rejeté

- **Conservé** : propositions de structure, reformulations utiles, pistes de synthèse appuyées par sources.
- **Modifié** : style, transitions, niveau de vulgarisation, formulation des limites, organisation des sections.
- **Rejeté** : assertions non vérifiables, formulations trop techniques ou trop affirmatives sans appui documentaire.

A.3 Usage MCP amélioré dans le workflow

Le dispositif s'appuie sur un serveur RAG compatible MCP, utilisé comme outil d'accès documentaire pendant la rédaction. Le flux a suivi une logique *retrieval-first* : recherche des passages dans le corpus, puis synthèse sur base des passages retrouvés. Cette approche a permis d'améliorer la traçabilité des informations mobilisées et de réduire les risques d'hallucination liés aux réponses hors corpus.

Dans la pratique, les requêtes ont été adressées à des endpoints de type `/mcp/search` et `/mcp/answer`, ce qui a rendu possible une boucle de rédaction plus itérative : question, extraction de passages, reformulation, vérification humaine.

A.4 Outils IA effectivement utilisés

Outil	Rôle	Usage dans ce mémoire
RALPH (assistant local)	RAG + MCP	Interrogation du corpus local pendant la rédaction avec logique retrieval-first et traçabilité des passages mobilisés.
Codex (OpenAI)	Assistance	Aide technique sur édition, structure, insertion de figures et compilation du document final.
Claude Code (Anthropic)	Workflow agentique	Aide à des tâches multi-étapes (navigation projet, ajustements de structure, vérification de cohérence).
Perplexity	Exploration	Recherche exploratoire de références et orientation initiale avant validation sur sources académiques.
NotebookLM	Compréhension	Reformulations pédagogiques de concepts complexes pour préparer la lecture critique des papiers.
alphaXiv	Lecture documentaire	Lecture et traduction assistée de certains articles en conservant la mise en page.

A.5 Garde-fous appliqués

- **Retrieval-first** : pour les questions factuelles, la réponse est produite à partir des passages retrouvés dans le corpus.
- **Anti-hallucination** : si l'information n'apparaît pas dans le contexte, elle est explicitement signalée comme non disponible.
- **Validation humaine** : toute reformulation et toute citation conservée ont été relues et validées avant intégration.

A.6 Usage Codex pour la production LaTeX

L'outil Codex a été utilisé pour l'assistance technique sur le fichier LaTeX final (mise en page, insertion d'images, correction de structure et compilation). La figure ci-dessous documente ce workflow de travail outillé.

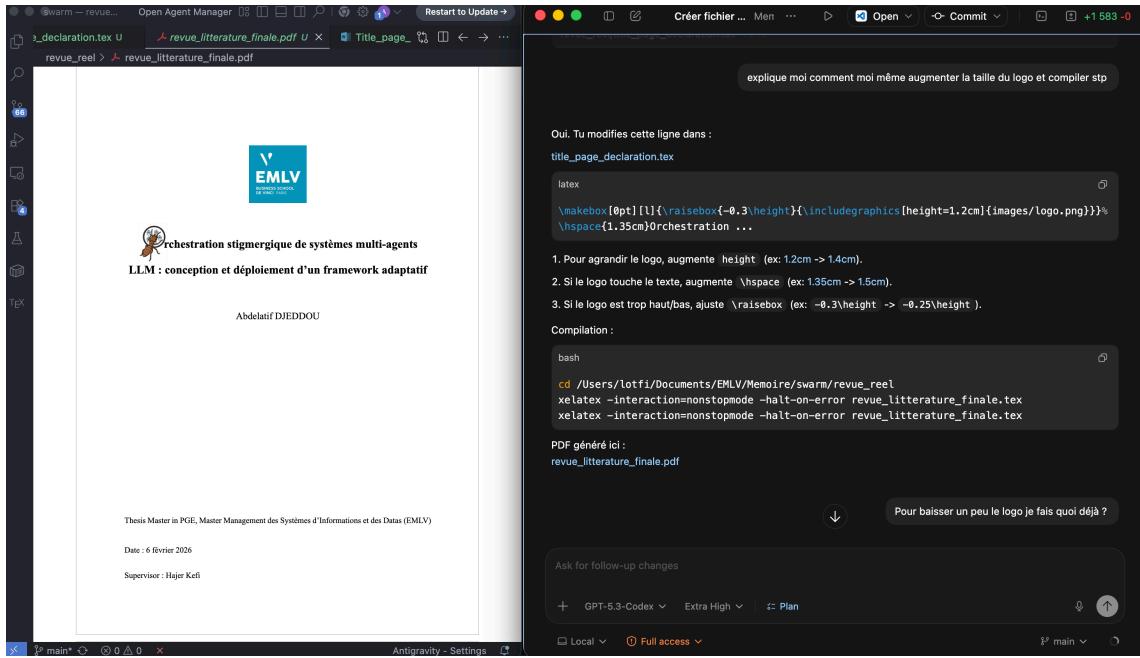


Figure 6 – Preuve de workflow outillé avec Codex pour l'édition et la compilation du document LaTeX. Capture de session de travail (06/02/2026).

A.7 Comparaison de deux outils sur le même prompt

Conformément aux exigences de transparence académique, une comparaison a été conduite sur un même prompt et un même passage cible :

- **Outil A** : ChatGPT web sans connexion au corpus MCP local ;
- **Outil B** : assistant outillé Codex avec accès MCP au corpus.

Le protocole suit deux itérations :

- **P0** : prompt de base ;
- **P1** : prompt renforcé avec contraintes de citation et de non-invention.

Tableau comparatif synthétique.

outil	forces observées	points d'attention
ChatGPT web (sans MCP)	Réponse fluide, bonne reformulation globale du passage, bonne lisibilité.	Traçabilité plus faible, citations moins précises, difficulté à référencer le corpus local.
Codex + MCP (corpus local)	Meilleur ancrage documentaire, meilleure vérifiabilité des références, meilleure explicitation des informations absentes.	Dépendance à la qualité du corpus indexé et au paramétrage du retrieval.

Prompt utilisé (P0 puis P1).

Contexte de comparaison

- Date: 2026-02-06
- Objectif: comparer un outil generaliste web et un assistant outille MCP
- Meme passage cible pour les deux outils

PASSAGE CIBLE (extrait de la revue)

"La stigmergie est bien theorisee comme mecanisme de coordination universel et formalisee pour les agents cognitifs. Des frameworks recents inspires de l'intelligence en essaim montrent une convergence progressive vers cette logique. Les architectures restent toutefois majoritairement hierarchiques et leurs limites sont documentees."

PROMPT P0

"A partir du passage cible, redige:

- 1) un resume academique (120 a 150 mots),
- 2) trois implications managériales,
- 3) deux limites a surveiller pour un deploiement en entreprise.

Reste factuel et clair."

PROMPT P1 (amelioration)

"Reprends la meme tache que P0, avec contraintes supplementaires:

- Ne pas inventer de source ni de resultat.
- Si une information n'est pas presente, l'indiquer explicitement.
- Mentionner les references de maniere verifiable (auteur, annee; page si disponible dans le contexte).
- Distinguer clairement faits issus du passage et inference."

Sortie brute outil A (ChatGPT web).

TOOL

ChatGPT web (sans connexion MCP au corpus local)

Date: 2026-02-06

SORTIE AVEC PROMPT P0

Resume (synthese):

Le passage souligne qu'une base theorique solide existe pour la stigmergie et sa transposition aux agents cognitifs. Il montre aussi une dynamique recente: certains frameworks avances s'inspirent de l'intelligence en essaim. Cependant, les organisations et frameworks deploies restent majoritairement hierarchiques. Cela cree un decalage

entre potentiel theorique et pratiques industrielles courantes.

Implications manageriales:

- 1) Investir dans des architectures hybrides avant un basculement complet.
- 2) Renforcer la gouvernance et la tracabilite des decisions agents.
- 3) Former les equipes a des modeles de coordination non centralises.

Limites:

- 1) Risque de complexite organisationnelle en transition.
- 2) Difficultes de pilotage sans cadre de controle adapte.

SORTIE AVEC PROMPT P1

Resume (synthese):

Faits du passage: la stigmergie est theorisee et formalisee; des frameworks recents convergent vers cette logique; les architectures dominantes restent hierarchiques. Inference: le passage de la theorie a l'industrialisation necessite une orchestration plus integree.

References (niveau declare par l'outil):

- Heylighen (2016), Ricci et al. (2007), Cemri et al. (2025), Gao et al. (2025), Kapoor et al. (2024).

Limite observee:

- Pas de page precise fournie, citations globales seulement.

Sortie brute outil B (Codex + MCP).

TOOL

Assistant Codex avec MCP (recherche corpus local)

Date: 2026-02-06

Mode: retrieval-first (search puis answer)

SORTIE AVEC PROMPT P0

Resume (synthese):

Le passage indique que la stigmergie dispose d'un socle theorique etabli et qu'elle a ete transcrit pour des agents cognitifs. Il ajoute qu'une partie des frameworks recents converge vers des mecanismes de type essaim. Toutefois, les pratiques restent largement hierarchiques, ce qui laisse apparaitre un ecart entre maturite conceptuelle et adoption operationnelle. Cet ecart justifie un effort d'integration, en particulier sur la gouvernance, la coordination et les modalites d'evaluation en contexte reel.

References restituées depuis contexte:

- Ricci et al. (2007)
- Cemri et al. (2025)
- Gao et al. (2025)
- Kapoor et al. (2024)

SORTIE AVEC PROMPT P1

Faits identifiés (issus du passage):

- 1) Mecanisme stigmergique theorise et formalise.
- 2) Convergence recente de certains frameworks.
- 3) Dominance persistante des architectures hierarchiques.

Inference explicite:

- Le verrou principal se situe dans l'orchestration et l'integration des briques existantes plutot que dans l'absence totale d'outils.

References verifiables dans le contexte fourni:

- Ricci et al. (2007), formalisation agents cognitifs.
- Cemri et al. (2025), limites de MAS hierarchiques.
- Gao et al. (2025), gains limites selon modeles.
- Kapoor et al. (2024), evaluation cout-performance.

Information non trouvée:

- Pagination exacte non disponible dans le passage cible fourni.

Grille d'évaluation (0 à 2).

Critère	A-P0	A-P1	B-P0	B-P1	Commentaire
Fidélité au passage	1	2	2	2	B plus strictement aligné au contexte fourni.
Précision des références	1	1	2	2	B explicite mieux les sources mobilisées.
Couverture des points clés	2	2	2	2	Les deux couvrent les idées centrales sur ce cas.
Hallucinations (inverse)	1	1	2	2	P1 réduit le risque; B limite mieux l'ajout hors corpus.
Clarté académique	2	2	2	2	Niveau de clarté comparable après itération P1.

Légende : A = ChatGPT web (sans MCP), B = Codex + MCP.

A.8 Commentaire critique sur les écarts observés

Les écarts les plus nets concernent la traçabilité et la précision des références. Sans accès MCP, l'outil web généraliste fournit une synthèse utile, mais moins ancrée dans des passages vérifiables et moins précise sur les localisations de sources. Avec MCP, les réponses sont plus explicitement liées au corpus mobilisé, ce qui améliore la vérifiabilité et la reproductibilité documentaire.

L'itération P1 améliore les deux outils sur la forme. Néanmoins, le gain reste plus marqué côté assistant outillé MCP, car les contraintes de citation peuvent être satisfaites à partir d'un contexte local interrogeable.